

Code Source de l'application SimpleTab

Romain Sauser
I.DA-P4A
CFPT INFORMATIQUE

24.05.2018

Table des matières

1	SimpleTab	3
1.1	config	3
1.1.1	conparam.php	3
1.2	controller	3
1.2.1	getUserByEmail.php	3
1.2.2	getUserByPseudo.php	4
1.2.3	getRateByTabId.php	4
1.2.4	deleteUserById.php	5
1.2.5	getInfosTab.php	5
1.2.6	getTabAndRelatedArtistByName.php	6
1.2.7	updateTabRate.php	7
1.2.8	approuveTab.php	7
1.2.9	getAllNonApprouvedTab.php	8
1.2.10	getUserById.php	8
1.2.11	identifyUser.php	9
1.2.12	getTabAndRelatedArtistPostedByUser.php	10
1.2.13	getUsersAndNbTabPosted.php	10
1.2.14	getRateByUserId.php	11
1.2.15	getArtistNames.php	11
1.2.16	getTabById.php	12
1.2.17	modifyTabById.php	12
1.2.18	getTabAndRelatedArtist.php	13
1.2.19	getTabByArtist.php	14
1.2.20	addTab.php	14
1.2.21	deleteTabById.php	15
1.2.22	getTabByTitle.php	16
1.2.23	addComment.php	17
1.2.24	getDifficultyInLetters.php	17
1.2.25	addUser.php	18
1.2.26	getCommentsByTab.php	19
1.2.27	addRate.php	19
1.2.28	destroySession.php	20
1.3	model	20
1.3.1	userManager.php	20
1.3.2	tablatureManager.php	23
1.3.3	commentManager.php	28
1.3.4	rateManager.php	30
1.3.5	artistManager.php	32
1.3.6	rate.php	33
1.3.7	tablature.php	34
1.3.8	user.php	36
1.3.9	comment.php	37

	1.3.10	Artist.php	39
	1.3.11	database.php	39
1.4	public	40
	1.4.1	css	40
	1.4.2	js	40
1.5	tabs	45
	1.5.1	81.php	45
	1.5.2	83.php	46
	1.5.3	84.php	47
1.6	view	49
	1.6.1	homePage.php	49
	1.6.2	search.php	51
	1.6.3	tablaturePage.php	54
	1.6.4	tablatureManagerPage.php	58
	1.6.5	modals.php	61
	1.6.6	tablatureAndUserManagerPage.php	63

Chapitre 1

SimpleTab

1.1 config

1.1.1 conparam.php

```
1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 08.05.2018
5  * projet: SimpleTab
6  * description: Variables pour la connection à la base de données
7  */
8 define ('EDB_DBTYPE', "mysql");
9 define ('EDB_HOST', "localhost");
10 define ('EDB_PORT', "3306");
11 define ('EDB_DBNAME', "simpletab");
12 define ('EDB_USER', "root");
13 define ('EDB_PASS', "root");
14
15 ?>
```

1.2 controller

1.2.1 getUserByEmail.php

```
1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 24.05.2018
5  * projet: SimpleTab
6  * description : récupère un utilisateur depuis son email
7  */
8
9 require_once '../model/userManager.php';
10
11 // Nécessaire lorsqu'on retourne du json
12 header('Content-Type: application/json');
13
14 // Je récupère l'id de l'utilisateur
15 $email = "";
16
17
18 if (isset($_POST['email']))
19 {
20     $email = $_POST['email'];
21 }
22
23 if ($email != ""){
24     $user = userManager::getInstance()->getUserByEmail($email);
25     if ($user === false){
26         return false; // le pseudo est libre
27         exit();
28     }
29
30     $jsn = json_encode($user);
31     // Problème d'encodage Json
32     if ($jsn == FALSE){
33         $code = json_last_error();
34         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json ('.$code.')"}';
35         exit();
36     }
37     // Succès
```

```

38     echo '{ "ReturnCode": 0, "Data": '.utf8_encode($jsn).'}';
39     exit();
40 }
41 }
42
43 // Erreur
44 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
45
46 ?>

```

1.2.2 getUserByPseudo.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 24.05.2018
5  * projet: SimpleTab
6  * description : Récupère l'utilisateur par son pseudo
7  */
8
9 require_once '../model/userManager.php';
10
11 // Nécessaire lorsqu'on retourne du json
12 header('Content-Type: application/json');
13
14 // Je récupère l'id de l'utilisateur
15 $pseudoUser = "";
16
17
18 if (isset($_POST['pseudo']))
19 {
20     $pseudoUser = $_POST['pseudo'];
21 }
22
23 if ($pseudoUser != ""){
24     $user = userManager::getInstance()->getUserByPseudo($pseudoUser);
25     if ($user === false){
26         return false; // le pseudo est libre
27         exit();
28     }
29
30     $jsn = json_encode($user);
31     // Problème d'encodage Json
32     if ($jsn == FALSE){
33         $code = json_last_error();
34         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json ('.$code.')"}';
35         exit();
36     }
37     // Succès
38     echo '{ "ReturnCode": 0, "Data": '.utf8_encode($jsn).'}';
39     exit();
40 }
41 }
42
43 // Erreur
44 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
45
46 ?>

```

1.2.3 getRateByTabId.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 23.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Récupère les informations des notes par l'id d'une tablature
11 */
12
13
14
15
16 require_once "../model/rateManager.php";
17
18 // Je récupère l'id de la tablature
19 $idTab = -1;
20
21
22 if (isset($_POST['idTab']))
23 {
24     $idTab = $_POST['idTab'];
25 }

```

```

26
27 if ($idTab != -1) {
28
29     $rate = rateManager::getInstance()->getRateByTabId($idTab);
30 }
31 if ($rate === false){
32     echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
33     exit();
34 }
35
36 $jsn = json_encode($rate);
37 // Problème d'encodage Json
38 if ($jsn == FALSE){
39     $code = json_last_error();
40     echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")}';
41     exit();
42 }
43 // Succès
44 echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
45 exit();
46
47
48 ?>

```

1.2.4 deleteUserById.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8
9 /**
10 * @copyright romain.ssr@eduge.ch 2018
11 * @brief Supprime l'utilisateur à partir d'un ID
12 */
13
14 require_once '../model/userManager.php';
15
16 // Nécessaire lorsqu'on retourne du json
17 header('Content-Type: application/json');
18
19 // Je récupère l'id de l'utilisateur
20 $idUser = -1;
21
22
23 if (isset($_POST['idUser']))
24 {
25     $idUser = $_POST['idUser'];
26 }
27
28
29 if ($idUser != -1){
30     $success = userManager::getInstance()->deleteUserById($idUser);
31     if ($success === false){
32         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
33         exit();
34     }
35
36     $jsn = json_encode($success);
37     // Problème d'encodage Json
38     if ($jsn == FALSE){
39         $code = json_last_error();
40         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")}';
41         exit();
42     }
43     // Succès
44     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
45     exit();
46 }
47
48 // Erreur
49 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
50
51
52 ?>

```

1.2.5 getInfosTab.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab

```

```

6  */
7
8  /**
9   * @copyright romain.ssr@eduge.ch 2018
10  * @brief Récupère les informations des tablatures au format XML
11  */
12
13 session_start();
14
15
16 require_once "../model/tablatuManager.php";
17
18 // Je récupère l'id de la tablature
19 $idTab = -1;
20
21
22 if (isset($_POST['idTab']))
23 {
24     $idTab = $_POST['idTab'];
25     $_SESSION['currentIdTab'] = $idTab;
26 }
27
28 if ($idTab != -1){
29
30     require_once "../tabs/" . $idTab . ".php";
31
32     $tabs = new SimpleXMLElement($xmlstr);
33     $title = $tabs->metadata->title;
34     $author = $tabs->metadata->author;
35     $tuning = $tabs->metadata->tuning;
36     $capo = $tabs->metadata->capo;
37     $key = $tabs->metadata->key;
38     $lvl = $tabs->metadata->level;
39     $bodyTab = $tabs->corpse;
40
41     $jsn = json_encode($tabs);
42     // Problème d'encodage Json
43     if ($jsn == FALSE){
44         $code = json_last_error();
45         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
46         exit();
47     }
48     // Succès
49     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
50     exit();
51 }
52 }
53
54 ?>

```

1.2.6 getTabAndRelatedArtistByName.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 17.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10  * @brief Récupère les tablatures ainsi que leur artiste posté par un utilisateur par leurs nom.
11  */
12
13 require_once '../model/tablatuManager.php';
14
15 // Nécessaire lorsqu'on retourne du json
16 header('Content-Type: application/json');
17
18 $tabTitleOrArtistName="";
19
20 if(isset($_POST['tabTitleOrArtistName']))
21 {
22     $tabTitleOrArtistName = $_POST['tabTitleOrArtistName'];
23 }
24
25 if($tabTitleOrArtistName!="")
26 {
27     $tablature = TablatuManager::getInstance()->getTabAndRelatedArtistByName($tabTitleOrArtistName);
28 }
29
30 if ($tablature === false){
31     echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
32     exit();
33 }
34
35 $jsn = json_encode($tablature);
36 // Problème d'encodage Json
37 if ($jsn == FALSE){
38     $code = json_last_error();

```

```

39     echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
40     exit();
41 }
42 // Succès
43 echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
44 exit();
45
46
47 ?>

```

1.2.7 updateTabRate.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Récupère les informations des tablatures au format XML
11 */
12
13
14
15 require_once "../model/tablaturationManager.php";
16
17 // Je récupère l'id de la tablature
18 $idTab = -1;
19
20
21 if (isset($_POST['idTab']))
22 {
23     $idTab = $_POST['idTab'];
24 }
25
26 if ($idTab != -1){
27     $success = tablaturationManager::getInstance()->updateTabRate($idTab);
28
29     $jsn = json_encode($tabs);
30     // Problème d'encodage Json
31     if ($jsn == FALSE){
32         $code = json_last_error();
33         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
34         exit();
35     }
36     // Succès
37     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
38     exit();
39 }
40
41 }
42
43 ?>

```

1.2.8 approuveTab.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Met l'état approuvé d'une tablature à true
11 */
12
13 require_once '../model/tablaturationManager.php';
14
15 // Nécessaire lorsqu'on retourne du json
16 header('Content-Type: application/json');
17
18 // Je récupère l'id de la tablature
19 $idTab = -1;
20
21
22 if (isset($_POST['idTab']))
23 {
24     $idTab = $_POST['idTab'];
25 }
26
27
28 if ($idTab != -1){

```



```

29 $success = TablatureManager::getInstance()->approuveTab($idTab);
30 if ($success === false){
31     echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
32     exit();
33 }
34
35 $jsn = json_encode($success);
36 // Problème d'encodage Json
37 if ($jsn == FALSE){
38     $code = json_last_error();
39     echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
40     exit();
41 }
42 // Succès
43 echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
44 exit();
45
46 }
47
48 // Erreur
49 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
50
51 ?>

```

1.2.9 getAllNonApprovedTab.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Récupère toutes les tablatures dont le champs approved = 0
11 */
12
13
14 require_once '../model/tablatureManager.php';
15
16 // Nécessaire lorsqu'on retourne du json
17 header('Content-Type: application/json');
18
19
20 $tablature = TablatureManager::getInstance()->getAllNonApprovedTab();
21 if ($tablature === false){
22     echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
23     exit();
24 }
25
26 $jsn = json_encode($tablature);
27 // Problème d'encodage Json
28 if ($jsn == FALSE){
29     $code = json_last_error();
30     echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
31     exit();
32 }
33 // Succès
34 echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
35 exit();
36
37
38 ?>

```

1.2.10 getUserById.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Récupère l'utilisateur depuis son id
11 */
12
13 require_once '../model/userManager.php';
14
15 // Nécessaire lorsqu'on retourne du json
16 header('Content-Type: application/json');
17
18 // Je récupère l'id de l'utilisateur
19 $idUser = "";

```

```

20
21
22 if (isset($_POST['idUser']))
23 {
24     $idUser = $_POST['idUser'];
25 }
26
27 if ($idUser != ""){
28     $user = userManager::getInstance()->getUserById($idUser);
29     if ($user === false){
30         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
31         exit();
32     }
33
34     $jsn = json_encode($user);
35     // Problème d'encodage Json
36     if ($jsn == FALSE){
37         $code = json_last_error();
38         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json ('.$code.')"}';
39         exit();
40     }
41     // Succès
42     echo '{ "ReturnCode": 0, "Data": '.utf8_encode($jsn).'}';
43     exit();
44 }
45
46 // Erreur
47 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
48
49 ?>

```

1.2.11 identifyUser.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 16.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Identifie un utilisateur depuis son pseudo ou email et son mot de passe
11 */
12
13 session_start();
14
15 require_once '../model/userManager.php';
16
17 // Nécessaire lorsqu'on retourne du json
18 header('Content-Type: application/json');
19
20 // Je récupère les champs dont j'ai besoin
21 $mailOrPseudoUser = "";
22 $passwordUser = "";
23
24 if (isset($_POST['mailOrPseudo']) && isset($_POST['pwdConnexion']))
25 {
26     $mailOrPseudoUser = $_POST['mailOrPseudo'];
27     $passwordUser = $_POST['pwdConnexion'];
28 }
29
30 if ($mailOrPseudoUser != "" || $passwordUser != ""){
31     $passwordHashed = userManager::getInstance()->getPasswordFromEmailOrPseudo($mailOrPseudoUser);
32     $passwordHashed = $passwordHashed[0]['pwdUser'];
33     $success = password_verify($passwordUser, $passwordHashed);
34     $user = userManager::getInstance()->identifyUser($mailOrPseudoUser);
35     if ($success === false){
36         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
37         exit();
38     }
39     // l'utilisateur est identifié et sa session est créée
40     $_SESSION['user'] = $user;
41
42     $jsn = json_encode($user);
43     // Problème d'encodage Json
44     if ($jsn == FALSE){
45         $code = json_last_error();
46         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json ('.$code.')"}';
47         exit();
48     }
49     // Succès
50     echo '{ "ReturnCode": 0, "Data": '.utf8_encode($jsn).'}';
51     exit();
52 }
53
54 // Erreur
55 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';

```

```
57
58 ?>
```

1.2.12 getTabAndRelatedArtistPostedByUser.php

```
1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 16.05.2018
5  * projet: SimpleTab
6  */
7
8
9 /**
10 * @copyright romain.ssr@eduge.ch 2018
11 * @brief Récupère les tablatures ainsi que leur artiste posté par un utilisateur
12 */
13
14 require_once '../model/tablatureManager.php';
15
16 // Nécessaire lorsqu'on retourne du json
17 header('Content-Type: application/json');
18
19 $userId="";
20
21 if(isset($_POST['idUser']))
22 {
23     $userId = $_POST['idUser'];
24 }
25
26 if($userId!="")
27 {
28     $tablature = TablatureManager::getInstance()->getTabAndRelatedArtistPostedByUser($userId);
29 }
30
31 if ($tablature === false){
32     echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
33     exit();
34 }
35
36 $jsn = json_encode($tablature);
37 // Problème d'encodage Json
38 if ($jsn == FALSE){
39     $code = json_last_error();
40     echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
41     exit();
42 }
43 // Succès
44 echo '{ "ReturnCode": 0, "Data": \' .utf8_encode($jsn).\' }';
45 exit();
46
47
48 ?>
```

1.2.13 getUsersAndNbTabPosted.php

```
1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8
9 /**
10 * @copyright romain.ssr@eduge.ch 2018
11 * @brief Récupère les utilisateurs et le nombre de tablatures qu'ils ont posté
12 */
13
14 require_once '../model/userManager.php';
15
16 // Nécessaire lorsqu'on retourne du json
17 header('Content-Type: application/json');
18
19
20 $user = userManager::getInstance()->getUsersAndNbTabPosted();
21 if ($user === false){
22     echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
23     exit();
24 }
25
26 $jsn = json_encode($user);
27 // Problème d'encodage Json
28 if ($jsn == FALSE){
29     $code = json_last_error();
30     echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
```

```

31     exit();
32 }
33 // Succès
34 echo '{ "ReturnCode": 0, "Data": ' . utf8_encode($jsn) . ' }';
35 exit();
36
37
38 ?>

```

1.2.14 getRateByUserId.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 23.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Récupère les informations des notes par l'id d'un utilisateur
11 * */
12
13
14
15
16 require_once "../model/rateManager.php";
17
18 // Je récupère l'id de l'utilisateur
19 $idUser = -1;
20
21
22 if (isset($_POST['idUser']))
23 {
24     $idUser = $_POST['idUser'];
25 }
26
27 if ($idUser != -1) {
28
29     $rate = rateManager::getInstance()->getRateByUserId($idUser);
30 }
31 if ($rate === false){
32     echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données" }';
33     exit();
34 }
35
36 $jsn = json_encode($rate);
37 // Problème d'encodage Json
38 if ($jsn == FALSE){
39     $code = json_last_error();
40     echo '{ "ReturnCode": 3, "Message": "Un problème de d'encodage json (' . $code . ')" }';
41     exit();
42 }
43 // Succès
44 echo '{ "ReturnCode": 0, "Data": ' . utf8_encode($jsn) . ' }';
45 exit();
46
47
48 ?>

```

1.2.15 getArtistNames.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Récupère tous les noms des artistes
11 * */
12
13
14 require_once '../model/artistManager.php';
15
16 // Nécessaire lorsqu'on retourne du json
17 header('Content-Type: application/json');
18
19
20 $artist = artistManager::getInstance()->getArtist();
21 if ($artist === false){
22     echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données" }';
23     exit();
24 }

```

```

25
26 $jsn = json_encode($artist);
27 // Problème d'encodage Json
28 if ($jsn == FALSE){
29     $code = json_last_error();
30     echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")';
31     exit();
32 }
33 // Succès
34 echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
35 exit();
36
37
38 ?>

```

1.2.16 getTabById.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 23.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Récupère une tablatures par son id
11 */
12
13 require_once '../model/tablaturationManager.php';
14
15 // Nécessaire lorsqu'on retourne du json
16 header('Content-Type: application/json');
17
18 // Je récupère le nom de l'artiste
19 $idTab = "";
20
21
22 if (isset($_POST['idTab']))
23 {
24     $idTab = $_POST['idTab'];
25 }
26
27
28 if ($idTab != ""){
29     $tablature = TablatureManager::getInstance()->getTabById($idTab);
30     if ($tablature == false){
31         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
32         exit();
33     }
34
35     $jsn = json_encode($tablature);
36     // Problème d'encodage Json
37     if ($jsn == FALSE){
38         $code = json_last_error();
39         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")';
40         exit();
41     }
42     // Succès
43     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
44     exit();
45 }
46
47
48 // Erreur
49 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
50
51 ?>

```

1.2.17 modifyTabById.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Modifie la tablatures à partir d'un ID
11 */
12
13 session_start();
14
15 require_once '../model/tablaturationManager.php';
16 require_once '../model/artistManager.php';

```

```

16
17
18 // Nécessaire lorsqu'on retourne du json
19 header('Content-Type: application/json');
20
21 // Je récupère l'id de la tablature
22 $idTab = -1;
23 $title = "";
24 $nameArtist="";
25 $lvl="";
26 $capo = "";
27 $key = "";
28 $tuning = "";
29 $tabBody="";
30
31 if (isset($_POST['modifyTitle']) && isset($_POST['modifyAuthor']) && isset($_POST['modifyLvl']) && ←
    isset($_POST['modifyCapo']) && isset($_POST['modifyKey']) && isset($_POST['modifyTuning']) && ←
    isset($_POST['modifyTabBody'])&& isset($_SESSION['currentIdTab']))
32 {
33     $title = $_POST['modifyTitle'];
34     $nameArtist=$_POST['modifyAuthor'];
35     $lvl=$_POST['modifyLvl'];
36     $capo = $_POST['modifyCapo'];
37     $key = $_POST['modifyKey'];
38     $tuning = $_POST['modifyTuning'];
39     $tabBody=$_POST['modifyTabBody'];
40     $idTab = $_SESSION['currentIdTab'];
41 }
42
43
44 if ($title != "" || $nameArtist != "" || $lvl != "" || $tuning != "" || $tabBody != "" || $idTab!=-1 || ←
    $capo != "" || $key!="|| $tabBody !="" || $idTab!=-1 ||$rateValue !=-1) {
45
46
47     // Récupère le nom de l'artiste dans la base sinon le crée en base et le récupère.
48     $artist = artistManager::getInstance()->getArtistByName($nameArtist);
49     if ($artist == false) {
50         if (!$artistAdded = artistManager::getInstance()->addArtist($nameArtist)) {
51             exit();
52         } else {
53             $artist = artistManager::getInstance()->getArtistByName($nameArtist);
54         }
55     }
56
57     $idArtist = $artist['0']['idArtist'];
58     $artistName = $artist['0']['nameArtist'];
59     $success = ←
    TablatureManager::getInstance()->modifyTab($idTab,$title,$artistName,$idArtist,$tuning,$capo,$key,$lvl,$tabBody);
60     if ($success === false){
61         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
62         exit();
63     }
64
65     $jsn = json_encode($success);
66     // Problème d'encodage Json
67     if ($jsn == FALSE){
68         $code = json_last_error();
69         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\'")';
70         exit();
71     }
72     // Succès
73     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
74     exit();
75 }
76
77 // Erreur
78 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
79
80
81 ?>

```

1.2.18 getTabAndRelatedArtist.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 15.05.2018
5  * projet: SimpleTab
6  */
7
8
9 /**
10 * @copyright romain.ssr@eduge.ch 2018
11 * @brief Récupère les tablatures et les artistes
12 */
13
14 require_once '../model/tablatureManager.php';
15
16 // Nécessaire lorsqu'on retourne du json
17 header('Content-Type: application/json');

```

```

18
19
20     $tablature = TablatureManager::getInstance()->getTabAndRelatedArtist();
21     if ($tablature === false){
22         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
23         exit();
24     }
25
26     $jsn = json_encode($tablature);
27     // Problème d'encodage Json
28     if ($jsn == FALSE){
29         $code = json_last_error();
30         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'$code.\")'}';
31         exit();
32     }
33     // Succès
34     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
35     exit();
36
37
38 ?>

```

1.2.19 getTabByArtist.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 15.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Récupère les tablatures d'un artiste en particulier
11 */
12
13 require_once '../model/tablatureManager.php';
14
15 // Nécessaire lorsqu'on retourne du json
16 header('Content-Type: application/json');
17
18 // Je récupère le nom de l'artiste
19 $artistName = "";
20
21
22 if (isset($_POST['artistName']))
23 {
24     $artistName = $_POST['artistName'];
25 }
26
27
28 if ($artistName != ""){
29     $tablature = TablatureManager::getInstance()->sortTabByArtist($artistName);
30     if ($tablature === false){
31         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
32         exit();
33     }
34
35     $jsn = json_encode($tablature);
36     // Problème d'encodage Json
37     if ($jsn == FALSE){
38         $code = json_last_error();
39         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'$code.\")'}';
40         exit();
41     }
42     // Succès
43     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
44     exit();
45 }
46
47 // Erreur
48 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
49
50 ?>
51

```

1.2.20 addTab.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 18.05.2018
5  * projet: SimpleTab
6  */
7
8 /**

```

```

9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Ajoute une tablature en base
11 */
12
13 session_start();
14
15 require_once '../model/tablatuManager.php';
16 require_once '../model/artistManager.php';
17
18 // Nécessaire lorsqu'on retourne du json
19 header('Content-Type: application/json');
20
21 // Je récupère les champs dont j'ai besoin
22 $title = "";
23 $nameArtist = "";
24 $lvl = "";
25 $capo = "";
26 $key = "";
27 $tuning = "";
28 $tabBody = "";
29
30
31 if (isset($_POST['addTitle']) && isset($_POST['addArtist']) && isset($_POST['addLvl']) && ←
    isset($_POST['addTuning']) && isset($_POST['addTabBody']) && isset($_SESSION['user']) )
32 {
33     $title = $_POST['addTitle'];
34     $nameArtist = $_POST['addArtist'];
35     $lvl = $_POST['addLvl'];
36     $tuning = $_POST['addTuning'];
37     $tabBody = $_POST['addTabBody'];
38     $userId = $_SESSION['user'][0]['idUsers'];
39 }
40
41 if(!isset($_POST['addCapo']))
42 {
43     $capo = "";
44 }
45
46
47 if(!isset($_POST['addKey']))
48 {
49     $key = "";
50 }
51
52
53 if ($title != "" || $nameArtist != "" || $lvl != "" || $tuning != "" || $tabBody != "" ){
54
55     // Récupère le nom de l'artiste dans la base sinon le crée en base et le récupère.
56     $artist = artistManager::getInstance()->getArtistByName($nameArtist);
57     if($artist == false)
58     {
59         if(!$artistAdded = artistManager::getInstance()->addArtist($nameArtist))
60         {
61             exit();
62         }
63         else
64         {
65             $artist = artistManager::getInstance()->getArtistByName($nameArtist);
66         }
67     }
68
69     $idArtist = $artist['0']['idArtist'];
70     $artistName = $artist['0']['nameArtist'];
71     $success = tablatuManager::getInstance()->addTab($title, $lvl, $idArtist, $userId, $artistName, ←
        $capo, $key, $tuning, $tabBody);
72     if ($success == false) {
73         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
74         exit();
75     }
76
77
78     $jsn = json_encode($success);
79     // Problème d'encodage Json
80     if ($jsn == FALSE){
81         $code = json_last_error();
82         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
83         exit();
84     }
85     // Succès
86     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
87     exit();
88 }
89 }
90
91 // Erreur
92 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
93
94 ?>

```

1.2.21 deleteTabById.php


```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8
9 /**
10 * @copyright romain.ssr@eduge.ch 2018
11 * @brief Supprime la tablatures à partir d'un ID
12 */
13
14 require_once '../model/tablaturationManager.php';
15
16 // Nécessaire lorsqu'on retourne du json
17 header('Content-Type: application/json');
18
19 // Je récupère l'id de la tablature
20 $idTab = -1;
21
22
23 if (isset($_POST['idTab']))
24 {
25     $idTab = $_POST['idTab'];
26 }
27
28
29 if ($idTab != -1){
30     $success = TablaturationManager::getInstance()->deleteTab($idTab);
31     if ($success === false){
32         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
33         exit();
34     }
35
36     $json = json_encode($success);
37     // Problème d'encodage Json
38     if ($json == FALSE){
39         $code = json_last_error();
40         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
41         exit();
42     }
43     // Succès
44     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($json).\'}';
45     exit();
46 }
47
48 // Erreur
49 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
50
51
52 ?>

```

1.2.22 getTabByTitle.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 17.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9 * @copyright romain.ssr@eduge.ch 2018
10 * @brief Récupère les tablatures à partir du titre
11 */
12
13 require_once '../model/tablaturationManager.php';
14
15 // Nécessaire lorsqu'on retourne du json
16 header('Content-Type: application/json');
17
18 // Je récupère le nom de l'artiste
19 $titleTab = "";
20
21
22 if (isset($_POST['titleTab']))
23 {
24     $titleTab = $_POST['titleTab'];
25 }
26
27
28 if ($titleTab != ""){
29     $tablature = TablaturationManager::getInstance()->getTabByTitle($titleTab);
30     if ($tablature === false){
31         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
32         exit();
33     }
34
35     $json = json_encode($tablature);

```

```

35     // Problème d'encodage Json
36     if ($jsn == FALSE){
37         $code = json_last_error();
38         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")';
39         exit();
40     }
41     // Succès
42     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
43     exit();
44 }
45
46 // Erreur
47 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
48
49 ?>

```

1.2.23 addComment.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 18.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Ajoute un commentaire en base
11 */
12
13 require_once '../model/commentManager.php';
14
15 // Nécessaire lorsqu'on retourne du json
16 header('Content-Type: application/json');
17
18 // Je récupère les champs dont j'ai besoin
19 $contentComment = "";
20 $idTabComment = -1;
21 $idUserComment = -1;
22
23
24
25
26 if (isset($_POST['contentComment']) && isset($_POST['idTabComment']) && isset($_POST['idUserComment'])) {
27     $contentComment = $_POST['contentComment'];
28     $idTabComment = $_POST['idTabComment'];
29     $idUserComment = $_POST['idUserComment'];
30 }
31
32
33 if ($contentComment != "" || $idTabComment != -1 || $idUserComment != -1 ){
34     $success = commentManager::getInstance()->addComment($contentComment,$idTabComment,$idUserComment);
35     if ($success == false){
36         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
37         exit();
38     }
39
40     $jsn = json_encode($success);
41     // Problème d'encodage Json
42     if ($jsn == FALSE){
43         $code = json_last_error();
44         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")';
45         exit();
46     }
47     // Succès
48     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
49     exit();
50 }
51
52 // Erreur
53 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
54
55 ?>

```

1.2.24 getDifficultyInLetters.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 15.05.2018
5  * projet: SimpleTab
6  */
7
8 /**

```

```

9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Renvoie la difficulté d'une tablature en toute lettres depuis un chiffre
11 */
12
13 require_once '../model/tablatuManager.php';
14
15 // Nécessaire lorsqu'on retourne du json
16 header('Content-Type: application/json');
17
18 // Je récupère le nom de l'artiste
19 $lvlTab = "";
20
21
22 if (isset($_POST['lvlTab']))
23 {
24     $lvlTab = $_POST['lvlTab'];
25 }
26
27
28 if ($lvlTab != ""){
29     $difficulty = TablatuManager::getInstance()->getDifficultyInLetters($lvlTab);
30     if ($difficulty === false){
31         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
32         exit();
33     }
34
35     $json = json_encode($difficulty);
36     // Problème d'encodage Json
37     if ($json == FALSE){
38         $code = json_last_error();
39         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json ('.$code.')"}';
40         exit();
41     }
42     // Succès
43     echo '{ "ReturnCode": 0, "Data": '.utf8_encode($json).'}';
44     exit();
45 }
46
47 // Erreur
48 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
49
50
51 ?>

```

1.2.25 addUser.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 15.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * @copyright romain.ssr@eduge.ch 2018
10 * @brief Ajoute un utilisateur en base
11 */
12
13 require_once '../model/userManager.php';
14
15 // Nécessaire lorsqu'on retourne du json
16 header('Content-Type: application/json');
17
18 // Je récupère les champs dont j'ai besoin
19 $nameUser = "";
20 $forenameUser = "";
21 $passwordUser = "";
22 $emailUser = "";
23 $pseudoUser = "";
24
25
26
27 if (isset($_POST['name']) && isset($_POST['forename']) && isset($_POST['pseudo']) &&isset($_POST['email']) &←
    &&isset($_POST['password'])&&isset($_POST['passwordConfirm']) && $_POST['password'] == <←
    $_POST['passwordConfirm'])
28 {
29     if (filter_var($_POST['email'], FILTER_VALIDATE_EMAIL)) {
30         if (checkdnsrr(explode('@',$_POST['email'])[1], 'MX')) {
31             $nameUser = $_POST['name'];
32             $forenameUser = $_POST['forename'];
33             $passwordUser = password_hash($_POST['password'], PASSWORD_BCRYPT);
34             $emailUser = $_POST['email'];
35             $pseudoUser = $_POST['pseudo'];
36
37         }
38     }
39 }
40
41 }
42

```

```

43 if ($nameUser != "" || $forenameUser != "" || $passwordUser != "" || $emailUser != "" || $pseudoUser != ""){
44     $success = <-
45     UserManager::getInstance()->addUser($nameUser,$forenameUser,$passwordUser,$emailUser,$pseudoUser);
46     if ($success == false){
47         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
48         exit();
49     }
50     $jsn = json_encode($success);
51     // Problème d'encodage Json
52     if ($jsn == FALSE){
53         $code = json_last_error();
54         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\'")}';
55         exit();
56     }
57     // Succès
58     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
59     exit();
60 }
61 }
62 // Erreur
63 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
64 }
65 }
66 }

```

1.2.26 getCommentsByTab.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 17.05.2018
5  * projet: SimpleTab
6  */
7
8
9 /**
10 * @copyright romain.ssr@eduge.ch 2018
11 * @brief Récupère les commentaires à partir d'une tablature
12 */
13
14 require_once '../model/commentManager.php';
15
16 // Nécessaire lorsqu'on retourne du json
17 header('Content-Type: application/json');
18
19 // Je récupère l'id de la tablature
20 $idTab = "";
21
22
23 if (isset($_POST['idTab']))
24 {
25     $idTab = $_POST['idTab'];
26 }
27
28
29 if ($idTab != ""){
30     $comment = commentManager::getInstance()->getCommentAndUsersByTab($idTab);
31     if ($comment == false){
32         echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
33         exit();
34     }
35
36     $jsn = json_encode($comment);
37     // Problème d'encodage Json
38     if ($jsn == FALSE){
39         $code = json_last_error();
40         echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\'")}';
41         exit();
42     }
43     // Succès
44     echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($jsn).\'}';
45     exit();
46 }
47
48 // Erreur
49 echo '{ "ReturnCode": 1, "Message": "Il manque les paramètres"}';
50 }
51 }
52 }

```

1.2.27 addRate.php

```

1 <?php
2
3 /**

```

```

4  * Auteur: romain.ssr@eduge.ch
5  * Date : 22.05.2018
6  * projet: SimpleTab
7  */
8
9  /**
10 * @copyright romain.ssr@eduge.ch 2018
11 * @brief Ajoute une note dans la base
12 */
13
14 require_once '../model/rateManager.php';
15 require_once '../model/tablatureManager.php';
16
17
18 // Nécessaire lorsqu'on retourne du json
19 header('Content-Type: application/json');
20
21 $idUser = -1;
22 $idTab = -1;
23 $rateValue = -1;
24
25 if(isset($_POST['idUser']) && isset($_POST['idTab']) && isset($_POST['rate']))
26 {
27     $idUser = $_POST['idUser'];
28     $idTab = $_POST['idTab'];
29     $rateValue = $_POST['rate'];
30 }
31
32 if($idUser != -1 && $idTab != -1 && $rateValue != -1)
33 {
34     $rate = rateManager::getInstance()->addRate($rateValue,$idTab,$idUser);
35     $averageRate = rateManager::getInstance()->getAverageRateByTabId($idTab);
36     $averageRate = $averageRate[0][0];
37     $updateRateSuccess = tablatureManager::getInstance()->updateTabRate($averageRate,$idTab);
38 }
39
40 if ($rate === false && $updateRateSuccess === false){
41     echo '{ "ReturnCode": 2, "Message": "Un problème de récupération des données"}';
42     exit();
43 }
44
45 $json = json_encode($rate);
46 // Problème d'encodage Json
47 if ($json == FALSE){
48     $code = json_last_error();
49     echo '{ "ReturnCode": 3, "Message": "Un problème de d\'encodage json (\'.$code.\")'}';
50     exit();
51 }
52 // Succès
53 echo '{ "ReturnCode": 0, "Data": \'.utf8_encode($json).\'}';
54 exit();
55
56
57 ?>

```

1.2.28 destroySession.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 16.05.2018
5  * projet: SimpleTab
6  */
7
8 //Permet de déconnecter l'utilisateur
9 session_start();
10 session_unset();
11 session_destroy();
12 header("Location: ../view/homePage.php")
13 ?>

```

1.3 model

1.3.1 userManager.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 15.05.2018
5  * projet: SimpleTab
6  */
7
8 require_once '../model/database.php';

```

```

9 require_once '../model/user.php';
10 require_once '../model/taBlatureManager.php';
11
12 /**
13  * @brief Helper class pour gérer les utilisateurs du site
14  *
15  * @author romain.ssr@eduge.ch
16  */
17 class UserManager
18 {
19     private static $objInstance;
20
21     /**
22      * @brief Constructeur de la Classe - Crée un nouveau UserManager si aucun n'existe déjà .
23      * Le met en privé de telle sorte à ce que personne ne peut créer un nouveau manager depuis ' = new
24      * UserManager();'
25      */
26     private function __construct ()
27     {
28         $this->users = array();
29     }
30
31     /**
32      * @brief Like the constructor, we make clone private so nobody can clone
33      * the instance
34      */
35     private function __clone ()
36     {}
37
38     /**
39      * @brief Retourne notre instance ou la crée
40      *
41      * @return $objInstance;
42      */
43     public static function getInstance ()
44     {
45         if (!self::$objInstance)
46         {
47             try
48             {
49                 self::$objInstance = new UserManager();
50             } catch (Exception $e)
51             {
52                 echo "EDataManager Error: " . $e;
53             }
54         }
55         return self::$objInstance;
56     }
57 }
58
59 /**
60  * Retourne le tableau de tous les utilisateurs
61  *
62  */
63 public function getUsers()
64 {
65     return $this->users;
66 }
67
68 /**
69  * @param $pseudo
70  * @return utilisateur sinon false
71  */
72 function getUserByPseudo($pseudo)
73 {
74     $db = Database::getInstance();
75
76     try {
77         $sql = $db->prepare("SELECT * FROM simpletab.users WHERE users.pseudoUser = :pseudo;");
78         $sql-> bindParam(':pseudo', $pseudo, PDO::PARAM_STR);
79         $sql->execute();
80         $result = $sql->fetchAll();
81         return $result;
82     }
83
84     catch (PDOException $e) {
85         return false;
86     }
87 }
88
89 /**
90  * @param $email
91  * @return utilisateur sinon false
92  */
93 function getUserByEmail($email)
94 {
95     $db = Database::getInstance();
96
97     try {
98         $sql = $db->prepare("SELECT * FROM simpletab.users WHERE users.emailUser = :email;");
99         $sql-> bindParam(':email', $email, PDO::PARAM_STR);
100         $sql->execute();
101         $result = $sql->fetchAll();
102         return $result;

```

```

103     }
104
105     catch (PDOException $e) {
106         return false;
107     }
108 }
109
110 /**
111  * @param $emailOrPseudo
112  * @return le mot de passe de l'utilisateur hashé si succès sinon false
113  */
114 function getPasswordFromEmailOrPseudo($mailOrPseudo)
115 {
116     $db = Database::getInstance();
117
118     try {
119         $sql = $db->prepare("SELECT users.pwdUser FROM simpletab.users WHERE (users.emailUser = ↵
:mailOrPseudo OR users.pseudoUser = :mailOrPseudo);");
120         $sql->bindParam(':mailOrPseudo', $mailOrPseudo, PDO::PARAM_STR);
121         $sql->execute();
122         $result = $sql->fetchAll();
123         return $result;
124     }
125
126     catch (PDOException $e) {
127         return false;
128     }
129 }
130
131 /**
132  * Récupère les utilisateurs et le nombre de tablatures qu'ils ont postés
133  * @return les utilisateurs et le nombre de tablatures qu'ils ont postés, sinon false
134  */
135 function getUsersAndNbTabPosted()
136 {
137     $db = Database::getInstance();
138
139     try {
140         $sql = $db->prepare("SELECT users.idUsers, pseudoUser, emailUser, count(users_idUsers) as nbTab
FROM users LEFT JOIN tablatures ON (tablatures.users_idUsers = ↵
users.idUsers)
WHERE users.role_idrole = 0
GROUP BY users.idUsers, pseudoUser, emailUser
ORDER BY count(users_idUsers) DESC, pseudoUser;");
145         $sql->execute();
146         $result = $sql->fetchAll();
147         return $result;
148     }
149
150     catch (PDOException $e) {
151         return false;
152     }
153 }
154
155 /**
156  * Ajoute un utilisateur en base
157  * @param $name
158  * @param $forename
159  * @param $password
160  * @param $email
161  * @param $pseudo
162  * @return true si succès sinon false
163  */
164 public function addUser($name, $forename, $password, $email, $pseudo)
165 {
166     $db = Database::getInstance();
167     $defaultRole = 0;
168
169     try {
170         $sql = $db->prepare("INSERT INTO users ( nameUser, forenameUser, pwdUser, emailUser, ↵
pseudoUser,role_idrole) VALUES ( :name, :forname,:password, :email, :pseudo,:role);");
171         $sql->bindParam(':name', $name, PDO::PARAM_STR);
172         $sql->bindParam(':forename', $forename, PDO::PARAM_STR);
173         $sql->bindParam(':password', $password, PDO::PARAM_STR);
174         $sql->bindParam(':email', $email, PDO::PARAM_STR);
175         $sql->bindParam(':pseudo', $pseudo, PDO::PARAM_STR);
176         $sql->bindParam(':role', $defaultRole, PDO::PARAM_STR);
177         $sql->execute();
178         return true;
179     }
180
181     catch (PDOException $e) {
182         return false;
183     }
184 }
185
186 /**
187  * Efface un utilisateur en base ainsi que ses tablatures associées
188  * @param $idUser
189  * @return true si succès sinon false
190  */
191 function deleteUserById($idUser)
192 {
193     $db = Database::getInstance();
194
195     try {

```

```

194         $sql = $db->prepare("SELECT idTab FROM simpletab.tablatures JOIN simpletab.users ON ↵
195         tablatures.users_idUsers = users.idUsers where users.idUsers = :idUser;");
196         $sql->bindParam(':idUser', $idUser, PDO::PARAM_INT);
197         $sql->execute();
198         $result = $sql->fetchAll();
199         if($result!= "" && isset($result) && count($result) != 0)
200         {
201             for ($i =0; $i<count($result);$i++)
202             {
203                 $successDeleteTab = tablatureMAnager::getInstance()->deleteTab($result[$i]['idTab']);
204             }
205         }
206         else
207         {
208             $successDeleteTab = true;
209         }
210
211         if($successDeleteTab)
212         {
213             $sql = $db->prepare("DELETE FROM simpletab.users WHERE users.idUsers = :idUser");
214             $sql->bindParam(':idUser', $idUser, PDO::PARAM_INT);
215             $sql->execute();
216             return true;
217         }
218         else
219         {
220             return false;
221         }
222     }
223     catch (PDOException $e) {
224         return false;
225     }
226 }
227
228 /**
229  * Retourne l'utilisateur s'il existe en base sinon false.
230  *
231  */
232 function identifyUser($mailOrPseudo)
233 {
234     $db = Database::getInstance();
235
236     try {
237         $sql = $db->prepare("SELECT * FROM simpletab.users WHERE (users.emailUser = :mailOrPseudo OR ↵
238         users.pseudoUser = :mailOrPseudo);");
239         $sql-> bindParam(':mailOrPseudo',$mailOrPseudo,PDO::PARAM_STR);
240         $sql-> bindParam(':password',$password,PDO::PARAM_STR);
241         $sql->execute();
242         $result = $sql->fetchAll();
243         return $result;
244     }
245     catch (PDOException $e) {
246         return false;
247     }
248 }
249
250
251 /** Tableau d'utilisateurs */
252 private $users;
253 }
254 ?>

```

1.3.2 tablatureManager.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 15.05.2018
5  * projet: SimpleTab
6  */
7
8 require_once '../model/database.php';
9 require_once '../model/tablature.php';
10 require_once '../model/commentManager.php';
11 require_once '../model/rateManager.php';
12
13
14
15 /**
16  * @brief Helper class pour gérer les tablatures du site
17  *
18  * @author romain.ssr@eduge.ch
19  */
20 class TablatureManager
21 {
22     private static $objInstance;
23
24     /**

```



```

25  * @brief Class Constructor - Create a new ECommentManager if one doesn't exist
26  * Set to private so no-one can create a new instance via ' = new
27  * ECommentManager();'
28  */
29  private function __construct ()
30  {
31      $this->tablature = array();
32  }
33
34  /**
35  * @brief Like the constructor, we make clone private so nobody can clone
36  * the instance
37  */
38  private function __clone ()
39  {}
40
41  /**
42  * @brief Retourne notre instance ou la crée
43  *
44  * @return $objInstance;
45  */
46  public static function getInstance ()
47  {
48      if (!self::$objInstance)
49      {
50          try
51          {
52              self::$objInstance = new TablatureManager();
53          } catch (Exception $e)
54          {
55              echo "EDataManager Error: " . $e;
56          }
57      }
58      return self::$objInstance;
59  }
60
61  /**
62  * Retourne le tableau de toutes les tablatures
63  *
64  */
65  public function getTablature()
66  {
67      return $this->tablature;
68  }
69
70
71  function getTabById($idTab)
72  {
73      $db = Database::getInstance();
74
75      try {
76          $sql = $db->prepare("SELECT * FROM simpletab.tablatures WHERE tablatures.idTab = :idTab;");
77          $sql->bindParam("idTab", $idTab, PDO::PARAM_INT);
78          $sql->execute();
79          $result = $sql->fetchAll();
80          return $result;
81      }
82
83      catch (PDOException $e) {
84          return false;
85      }
86  }
87
88  /**
89  * Retourne Un tableau contenant les tablatures et les artistes associés ou false si une erreur est ↵
90  * survenue
91  *
92  */
93  function getTabAndRelatedArtist()
94  {
95      $db = Database::getInstance();
96
97      try {
98          $sql = $db->prepare("SELECT * FROM simpletab.tablatures JOIN simpletab.artists ON ↵
99          tablatures.ARTISTS_idArtist = artists.idArtist WHERE tablatures.approuved = 1;");
100          $sql->execute();
101          $result = $sql->fetchAll();
102          return $result;
103      }
104
105      catch (PDOException $e) {
106          return false;
107      }
108  }
109
110  /**
111  * Retourne Un tableau contenant les tablatures et les artistes associés que l'utilisateur a posté ou ↵
112  * false si une erreur est survenue
113  * @param $userId -> l'identifiant de l'utilisateur
114  */
115  function getTabAndRelatedArtistPostedByUser($userId)
116  {
117      $db = Database::getInstance();

```

```

116
117
118     try {
119         $sql = $db->prepare("SELECT * FROM simpletab.tablatures JOIN simpletab.artists ON ↔
120 tablatures.ARTISTS_idArtist = artists.idArtist WHERE (tablatures.users_idUsers = :userId AND ↔
121 tablatures.approuved = 1);");
122         $sql->bindParam(':userId', $userId, PDO::PARAM_INT);
123         $sql->execute();
124         $result = $sql->fetchAll();
125         return $result;
126     }
127
128     catch (PDOException $e) {
129         return false;
130     }
131 }
132
133 /**
134 * Récupère toutes les tablatures non approuvée (champ approuved =0)
135 * @return un tableau des tablatures non approuvées ou false si échoue
136 */
137 function getAllNonApprouvedTab()
138 {
139     $db = Database::getInstance();
140
141     try {
142         $sql = $db->prepare("SELECT * FROM simpletab.tablatures JOIN simpletab.artists ON ↔
143 tablatures.ARTISTS_idArtist = artists.idArtist WHERE tablatures.approuved = 0;");
144         $sql->execute();
145         $result = $sql->fetchAll();
146         return $result;
147     }
148
149     catch (PDOException $e) {
150         return false;
151     }
152 }
153
154 /**
155 * Retourne les tablatures associées à un artiste, sinon false
156 * @param $artistName
157 */
158 function sortTabByArtist($artistName)
159 {
160     $db = Database::getInstance();
161
162     try {
163         $sql = $db->prepare("SELECT * FROM simpletab.tablatures JOIN simpletab.artists ON ↔
164 tablatures.ARTISTS_idArtist = artists.idArtist WHERE artists.nameArtist = :nameArtist;");
165         $sql->bindParam(':nameArtist', $artistName, PDO::PARAM_STR);
166         $sql->execute();
167         $result = $sql->fetchAll();
168         return $result;
169     }
170
171     catch (PDOException $e) {
172         return false;
173     }
174 }
175
176 /**
177 * @param $tabTitleOrArtistName
178 * @return la tablature qui correspond au nom de l'artiste ou au titre de la tablature, sinon false
179 */
180 function getTabAndRelatedArtistByName($tabTitleOrArtistName)
181 {
182     $db = Database::getInstance();
183
184     try {
185         $sql = $db->prepare("SELECT * FROM simpletab.tablatures JOIN simpletab.artists ON ↔
186 tablatures.ARTISTS_idArtist = artists.idArtist
187 WHERE tablatures.titleTab = :tabTitleOrArtistName OR ↔
188 artists.nameArtist = :tabTitleOrArtistName;");
189         $sql->bindParam(':tabTitleOrArtistName', $tabTitleOrArtistName, PDO::PARAM_STR);
190         $sql->execute();
191         $result = $sql->fetchAll();
192         return $result;
193     }
194
195     catch (PDOException $e) {
196         return false;
197     }
198 }
199
200 /**
201 * @param $titleTab
202 * @return Un tableau contenant les informations de la tablature récupérée par son titre
203 */
204 function getTabByTitle($titleTab)
205 {
206     $db = Database::getInstance();
207
208     try {

```

```

204         $sql = $db->prepare("SELECT * FROM simpletab.tablatures WHERE tablatures.titleTab = :titleTab;");
205         $sql->bindParam(':titleTab', $titleTab, PDO::PARAM_STR);
206         $sql->execute();
207         $result = $sql->fetchAll();
208         return $result;
209     }
210
211     catch (PDOException $e) {
212         return false;
213     }
214 }
215
216 /**
217  * Ajoute une tablature en base et dans un fichier php au format XML
218  * @param $title
219  * @param $lvl
220  * @param $artistId
221  * @param $userId
222  * @param $artistName
223  * @param $capo
224  * @param $key
225  * @param $tuning
226  * @param $tabBody
227  * @return true si la tablature a été correctement ajoutée en base et que le fichier xml de la ↵
228  * tablature a bien été créé, false sinon
229 */
229 function addTab($title, $lvl, $artistId, $userId, $artistName, $capo, $key, $tuning, $tabBody)
230 {
231     $db = Database::getInstance();
232     $path = "temp.php";
233     try {
234         $db->beginTransaction();
235
236         $sql = $db->prepare("INSERT INTO simpletab.tablatures (titleTab, pathTab, lvlTab, ↵
237         ARTISTS_idArtist, users_idUsers, approved) VALUES (:title, :path, :lvl, :artistId, :idUser, 0);");
238         $sql->bindParam(':title', $title, PDO::PARAM_STR);
239         $sql->bindParam(':path', $path, PDO::PARAM_STR);
240         $sql->bindParam(':lvl', $lvl, PDO::PARAM_STR);
241         $sql->bindParam(':artistId', $artistId, PDO::PARAM_STR);
242         $sql->bindParam(':idUser', $userId, PDO::PARAM_STR);
243         $sql->execute();
244
245         $lastId = $db->lastInsertId();
246         $newPath = $lastId.".php";
247
248         $sql = $db->prepare("UPDATE simpletab.tablatures SET pathTab = :path WHERE idTab = :idTab;");
249         $sql->bindParam(':idTab', $lastId, PDO::PARAM_STR);
250         $sql->bindParam(':path', $newPath, PDO::PARAM_STR);
251         $sql->execute();
252
253         $newPath = "../tabs/$lastId.php";
254         $tab = fopen($newPath, 'a+');
255         $lvl = $this->getDifficultyInLetters($lvl);
256
257         $tabXml = '<?php
258 <?xml version = "1.0" encoding="UTF-8" standalone="yes" ?>
259 <tabs>
260     <metadata>
261         <title> '.$title.' </title>
262         <author> '.$artistName.' </author>
263         <tuning> '.$tuning.' </tuning>
264         <capo> '.$capo.' </capo>
265         <key> '.$key.' </key>
266         <level> '.$lvl.' </level>
267     </metadata>
268     <corpse>
269         '.$tabBody.'
270     </corpse>
271 </tabs>
272 XML;
273 ?>';
274
275         $tabXml = str_replace('&', '&amp;', $tabXml);
276         fwrite($tab, $tabXml);
277         fclose($tab);
278
279         $db->commit();
280         return true;
281     }
282
283     catch (PDOException $e) {
284         $db->rollBack();
285         return false;
286     }
287 }
288
289 /**
290  * Modifie une tablature en base ainsi que dans le XML
291  * @param $idTab
292  * @param $title
293  * @param $artistName
294  * @param $artistId
295  * @param $tuning

```

```

296 * @param $capo
297 * @param $key
298 * @param $lvl
299 * @param $tabBody
300 * @return true si succès sinon false
301 */
302 function modifyTab($idTab, $title, $artistName, $artistId, $tuning, $capo, $key, $lvl, $tabBody)
303 {
304     $db = Database::getInstance();
305
306     try {
307         $db->beginTransaction();
308
309         $sql = $db->prepare("UPDATE simpletab.tablatures SET titleTab = :title, lvlTab = :lvl, ←
ARTISTS_idArtist = :idArtist, approuved = 0 WHERE (idTab = :idTab);");
310         $sql->bindParam(':title', $title, PDO::PARAM_STR);
311         $sql->bindParam(':idArtist', $artistId, PDO::PARAM_STR);
312         $sql->bindParam(':lvl', $lvl, PDO::PARAM_INT);
313         $sql->bindParam(':idTab', $idTab, PDO::PARAM_STR);
314         $sql->execute();
315
316         $path = "../tabs/$idTab.php";
317
318         $tab = fopen($path, 'w');
319         $lvl = $this->getDifficultyInLetters($lvl);
320
321         $tabXml = '<?php
322 $xmlstr = <<<XML
323 <?xml version = "1.0" encoding="UTF-8" standalone="yes" ?>
324 <tabs>
325     <metadata>
326         <title> '.$title.' </title>
327         <author> '.$artistName.' </author>
328         <tuning> '.$tuning.' </tuning>
329         <capo> '.$capo.' </capo>
330         <key> '.$key.' </key>
331         <level> '.$lvl.' </level>
332     </metadata>
333     <corpse>
334         '.$tabBody.'
335     </corpse>
336 </tabs>
337 XML;
338 >?';
339
340         $tabXml = str_replace('&', '&amp;', $tabXml);
341         ftruncate($tab, 0);
342         fputs($tab, $tabXml);
343         fclose($tab);
344
345         $db->commit();
346         return true;
347     }
348
349     catch (PDOException $e) {
350         $db->rollBack();
351         return false;
352     }
353 }
354
355 /**
356 * Supprime une tablature de la base ainsi que son fichier XML
357 * @param $idTab
358 * @return bool
359 */
360 function deleteTab($idTab)
361 {
362     $db = Database::getInstance();
363     $path = "../tabs/$idTab.php";
364
365     try {
366         $db->beginTransaction();
367
368         commentManager::getInstance()->deleteCommentByTabId($idTab);
369         rateManager::getInstance()->deleteRateByIdTab($idTab);
370
371         $sql = $db->prepare("DELETE FROM simpletab.tablatures WHERE (idTab = :idTab);");
372         $sql->bindParam(':idTab', $idTab, PDO::PARAM_INT);
373         $sql->execute();
374         $result = unlink($path);
375         if($result)
376         {
377             $db->commit();
378             return true;
379         }
380         else
381         {
382             $db->rollBack();
383         }
384     }
385
386     catch (PDOException $e) {
387         $db->rollBack();
388         return false;

```

```

389     }
390 }
391
392 /**
393  * Passe le champ d'une tablature approuvée à 1 ce qui la rend visible pour les utilisateurs
394  * @param $idTab
395  * @return bool
396  */
397 function approuveTab($idTab)
398 {
399     $db = Database::getInstance();
400
401     try {
402         $sql = $db->prepare("UPDATE simpletab.tablatures SET approuvée = 1 WHERE (idTab = :idTab);");
403         $sql->bindParam(':idTab', $idTab, PDO::PARAM_INT);
404         $sql->execute();
405         return true;
406     }
407
408     catch (PDOException $e) {
409         return false;
410     }
411 }
412
413 function updateTabRate($averageRate, $idTab)
414 {
415     $db = Database::getInstance();
416
417     try {
418         $sql = $db->prepare("UPDATE simpletab.tablatures SET rateTab = :averageRate WHERE (idTab = :idTab);");
419         $sql->bindParam(':idTab', $idTab, PDO::PARAM_INT);
420         $sql->bindParam(':averageRate', $averageRate, PDO::PARAM_INT);
421         $sql->execute();
422         return true;
423     }
424
425     catch (PDOException $e) {
426         return false;
427     }
428 }
429
430 /**
431  * @param $lvl
432  * @return la difficulté en lettre à partir d'un chiffre
433  */
434 function getDifficultyInLetters($lvl)
435 {
436     switch ($lvl)
437     {
438         case "0":
439             return "Facile";
440             break;
441         case "1":
442             return "Moyen";
443             break;
444
445         case "2":
446             return "Difficile";
447             break;
448     }
449 }
450 }
451
452 /** Tableau de tablature */
453 private $tablature;
454 }

```

1.3.3 commentManager.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 17.05.2018
5  * projet: SimpleTab
6  */
7
8
9 require_once '../model/database.php';
10 require_once '../model/comment.php';
11
12 /**
13  * @brief Helper class pour gérer les tablatures du site
14  *
15  * @author romain.ssr@eduge.ch
16  */
17 class CommentManager
18 {
19     private static $objInstance;
20

```

```

21  /**
22  * @brief Class Constructor - Create a new ECommentManager if one doesn't exist
23  * Set to private so no-one can create a new instance via ' = new
24  * ECommentManager();'
25  */
26  private function __construct ()
27  {
28      $this->comments = array();
29  }
30
31  /**
32  * @brief Like the constructor, we make clone private so nobody can clone
33  * the instance
34  */
35  private function __clone ()
36  {}
37
38  /**
39  * @brief Retourne notre instance ou la crée
40  *
41  * @return $objInstance;
42  */
43  public static function getInstance ()
44  {
45      if (!self::$objInstance)
46      {
47          try
48          {
49              self::$objInstance = new CommentManager();
50          } catch (Exception $e)
51          {
52              echo "EDataManager Error: " . $e;
53          }
54      }
55      return self::$objInstance;
56  }
57
58
59
60  /**
61  * @param $idTab -> l'id de la tablature associée au commentaire
62  * @return les commentaires associés à la tablature sous forme de tableau ou false si une erreur survient
63  */
64  function getCommentAndUsersByTab($idTab)
65  {
66      $db = Database::getInstance();
67
68      try {
69          $sql = $db->prepare("SELECT * FROM simpletab.comments JOIN simpletab.users ON ↵
comments.users_idUsers = users.idUsers WHERE comments.tablatures_idTab = :idTab;");
70          $sql->bindParam(':idTab', $idTab, PDO::PARAM_INT);
71          $sql->execute();
72          $result = $sql->fetchAll();
73          return $result;
74      }
75
76      catch (PDOException $e) {
77          return false;
78      }
79  }
80
81  /**
82  * Ajoute un commentaire en base
83  * @param $contentComment
84  * @param $tabIdComment
85  * @param $userIdComment
86  * @return bool
87  */
88  function addComment($contentComment, $tabIdComment, $userIdComment)
89  {
90      $db = Database::getInstance();
91
92      try {
93          $sql = $db->prepare("INSERT INTO simpletab.comments ( contentComment, tablatures_idTab, ↵
users_idUsers) VALUES ( :contentComment, :idTab, :idUser);");
94          $sql->bindParam(':contentComment', $contentComment, PDO::PARAM_STR);
95          $sql->bindParam(':idTab', $tabIdComment, PDO::PARAM_INT);
96          $sql->bindParam(':idUser', $userIdComment, PDO::PARAM_INT);
97          $sql->execute();
98          return true;
99      }
100      catch (PDOException $e) {
101          return false;
102      }
103  }
104
105  /**
106  * Efface un commentaire par son Id
107  * @param $idComment
108  * @return bool
109  */
110  function deleteCommentByTabId($idTab)
111  {
112      $db = Database::getInstance();

```

```

113     try{
114         $sql = $db->prepare("DELETE FROM simpletab.comments WHERE comments.tablatures_idTab = :idTab");
115         $sql->bindParam(':idTab', $idTab, PDO::PARAM_INT);
116         $sql->execute();
117         return true;
118     }
119     catch (PDOException $e) {
120         return false;
121     }
122 }
123
124 /** Tableau de commentaires */
125 private $comments;
126
127 }

```

1.3.4 rateManager.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8 require_once '../model/database.php';
9 require_once '../model/rate.php';
10
11 /**
12  * @brief Helper class pour gérer les artistes du site
13  *
14  * @author romain.ssr@eduge.ch
15  */
16 class rateManager
17 {
18     private static $objInstance;
19
20     /**
21      * @brief Class Constructor - Create a new ECommentManager if one doesn't exist
22      * Set to private so no-one can create a new instance via ' = new
23      * ECommentManager();'
24      */
25     private function __construct()
26     {
27         $this->rate = array();
28     }
29
30     /**
31      * @brief Like the constructor, we make clone private so nobody can clone
32      * the instance
33      */
34     private function __clone()
35     {
36     }
37
38     /**
39      * @brief Retourne notre instance ou la crée
40      *
41      * @return $objInstance;
42      */
43     public static function getInstance()
44     {
45         if (!self::$objInstance) {
46             try {
47                 self::$objInstance = new rateManager();
48             } catch (Exception $e) {
49                 echo "EDataManager Error: " . $e;
50             }
51         }
52         return self::$objInstance;
53     }
54
55     /**
56      * @return le tableau des notes sinon false
57      */
58     function getRate()
59     {
60     }
61
62     $db = Database::getInstance();
63
64     try {
65         $sql = $db->prepare("SELECT * FROM simpletab.rates");
66         $sql->execute();
67         $result = $sql->fetchAll();
68         return $result;
69     }
70     catch (PDOException $e) {
71         return false;
72     }

```

```

73 }
74
75 /**
76  * Ajoute une note en base
77  * @param $rate
78  * @param $idTab
79  * @param $idUser
80  * @return true si succès, sinon false
81  */
82 function addRate($rate,$idTab,$idUser)
83 {
84     $db = Database::getInstance();
85
86     try {
87         $sql = $db->prepare("INSERT INTO simpletab.rates (rate,tablatures_idTab,users_idUsers) VALUES ( ←
:rate, :idTab, :idUser);");
88         $sql->bindParam(':rate', $rate, PDO::PARAM_INT);
89         $sql->bindParam(':idTab', $idTab, PDO::PARAM_INT);
90         $sql->bindParam(':idUser', $idUser, PDO::PARAM_INT);
91         $sql->execute();
92         return true;
93     }
94     catch (PDOException $e) {
95         return false;
96     }
97 }
98
99 /**
100  * Récupère la moyenne des notes par l'id d'une tablature
101  * @param $idTab
102  * @return les notes de la tablature si succès sinon false
103  */
104 function getRateByTabId($idTab)
105 {
106     $db = Database::getInstance();
107     try{
108         $sql = $db->prepare("SELECT * FROM simpletab.rates WHERE rates.tablatures_idTab = :idTab");
109         $sql->bindParam(':idTab', $idTab, PDO::PARAM_INT);
110         $sql->execute();
111         $result = $sql->fetchAll();
112         return $result;
113     }
114     catch (PDOException $e) {
115         return false;
116     }
117 }
118
119 /**
120  * Récupère la moyenne des notes par l'id d'un utilisateur
121  * @param $idUser
122  * @return les notes de la tablature si succès sinon false
123  */
124 function getRateByUserId($idUser)
125 {
126     $db = Database::getInstance();
127     try{
128         $sql = $db->prepare("SELECT * FROM simpletab.rates WHERE rates.users_idUsers = :idUser");
129         $sql->bindParam(':idUser', $idUser, PDO::PARAM_INT);
130         $sql->execute();
131         $result = $sql->fetchAll();
132         return $result;
133     }
134     catch (PDOException $e) {
135         return false;
136     }
137 }
138
139 /**
140  * Supprime une note par l'id d'une tablature
141  * @param $idTab
142  * @return true si succès sinon false
143  */
144 function deleteRateByIdTab($idTab)
145 {
146     $db = Database::getInstance();
147     try{
148         $sql = $db->prepare("DELETE FROM simpletab.rates WHERE rates.tablatures_idTab = :idTab");
149         $sql->bindParam(':idTab', $idTab, PDO::PARAM_INT);
150         $sql->execute();
151         return true;
152     }
153     catch (PDOException $e) {
154         return false;
155     }
156 }
157
158 function getAverageRateByTabId($idTab)
159 {
160     $db = Database::getInstance();
161     try{
162         $sql = $db->prepare("SELECT ROUND(AVG(rates.rate)) FROM simpletab.rates WHERE ←
rates.tablatures_idTab = :idTab");
163         $sql->bindParam(':idTab', $idTab, PDO::PARAM_INT);
164         $sql->execute();

```



```

165         $result = $sql->fetchAll();
166         return $result;
167     }
168     catch (PDOException $e) {
169         return false;
170     }
171 }
172
173 /**
174  * @var Tableau des artistes
175  */
176 private $rate;
177 }

```

1.3.5 artistManager.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 18.05.2018
5  * projet: SimpleTab
6  */
7
8 require_once '../model/database.php';
9 require_once '../model/artist.php';
10
11 /**
12  * @brief Helper class pour gérer les artistes du site
13  *
14  * @author romain.ssr@eduge.ch
15  */
16 class artistManager
17 {
18     private static $objInstance;
19
20     /**
21      * @brief Class Constructor - Create a new ECommentManager if one doesn't exist
22      * Set to private so no-one can create a new instance via ' = new
23      * ECommentManager();'
24      */
25     private function __construct()
26     {
27         $this->artist = array();
28     }
29
30     /**
31      * @brief Like the constructor, we make clone private so nobody can clone
32      * the instance
33      */
34     private function __clone()
35     {
36     }
37
38     /**
39      * @brief Retourne notre instance ou la crée
40      *
41      * @return $objInstance;
42      */
43     public static function getInstance()
44     {
45         if (!self::$objInstance) {
46             try {
47                 self::$objInstance = new artistManager();
48             } catch (Exception $e) {
49                 echo "EDataManager Error: " . $e;
50             }
51         }
52         return self::$objInstance;
53     }
54
55     /**
56      * @return le tableau des artistes sinon false
57      */
58     function getArtist()
59     {
60     }
61
62     $db = Database::getInstance();
63
64     try {
65         $sql = $db->prepare("SELECT artists.nameArtist FROM simpletab.artists");
66         $sql->bindParam(':nameArtist', $nameArtist, PDO::PARAM_STR);
67         $sql->execute();
68         $result = $sql->fetchAll();
69         return $result;
70     }
71     catch (PDOException $e) {
72         return false;
73     }
74 }

```

```

75
76 /**
77  * Ajoute un artiste en base depuis son nom
78  * @param $nameArtist
79  * @return true si réussi sinon false
80  */
81 function addArtist($nameArtist)
82 {
83     $db = Database::getInstance();
84
85     try {
86         $sql = $db->prepare("INSERT INTO simpletab.artists (nameArtist) VALUES ( :nameArtist);");
87         $sql->bindParam(':nameArtist', $nameArtist, PDO::PARAM_STR);
88         $sql->execute();
89         return true;
90     }
91     catch (PDOException $e) {
92         return false;
93     }
94 }
95
96 /**
97  * Récupère un artiste par son nom
98  * @param $nameArtist
99  * @return l'artiste si réussi sinon false
100 */
101 function getArtistByName($nameArtist)
102 {
103     $db = Database::getInstance();
104
105     try {
106         $sql = $db->prepare("SELECT * FROM simpletab.artists WHERE artists.nameArtist = :nameArtist;");
107         $sql->bindParam(':nameArtist', $nameArtist, PDO::PARAM_STR);
108         $sql->execute();
109         $result = $sql->fetchAll();
110         return $result;
111     }
112     catch (PDOException $e) {
113         return false;
114     }
115 }
116
117 /**
118  * @var Tableau des artistes
119  */
120 private $artist;
121 }

```

1.3.6 rate.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * Classe des artistes
10  * @author RS
11  */
12 class rate implements JsonSerializable
13 {
14     /**
15      * Rate constructor.
16      * @param int $InArtistId -> L'id de l'artiste
17      * @param string $InArtistName -> Le nom de l'artiste
18      */
19     public function __construct ($InRateId = - 1, $InRate = -1, $InRateIdTab = -1, $InRateIdUser = -1)
20     {
21         $this->id = $InRateId;
22         $this->rate = $InRate;
23         $this->idTab = $InRateIdTab;
24         $this->idUser = $InRateIdUser;
25     }
26
27     /**
28      * @brief On ne laisse pas cloner une tablature
29      */
30     private function __clone ()
31     {}
32
33     /**
34      * @brief Est-ce que cet objet est valide
35      *
36      * @return True si valide, autrement false
37      */
38     public function isValid ()
39     {

```

```

40     return ( $this->id == -1 || $this->rate == -1 || $this->idTab == -1 || $this->idUser == -1 ) ? false : ←
41     true;
42 }
43
44 /**
45  * @brief Getter
46  * @return L'identifiant de la note
47  */
48 public function getId ()
49 {
50     return $this->id;
51 }
52
53 /**
54  * @brief Getter
55  *
56  * @return La valeur de la note
57  */
58 public function getRate ()
59 {
60     return $this->rate;
61 }
62
63 /**
64  * @brief Getter
65  *
66  * @return L'id de la tablature associée
67  */
68 public function getIdTab ()
69 {
70     return $this->idTab;
71 }
72
73 /**
74  * @brief Getter
75  *
76  * @return L'id de l'utilisateur associé
77  */
78 public function getIdUser ()
79 {
80     return $this->idUser;
81 }
82
83
84 public function jsonSerialize()
85 {
86     return get_object_vars($this);
87 }
88
89 /** Identifiant de la note */
90 private $id;
91 /** La valeur de la note */
92 private $rate;
93 /** L'id de la tablature associée */
94 private $idTab;
95 /** L'id de l'utilisateur associé */
96 private $idUser;
97 }

```

1.3.7 tablature.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 15.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * Classe des tablatures
10  * @author RS
11  */
12 class Tablature implements JsonSerializable
13 {
14     /**
15      * @brief Class Constructor
16      * @param $InTabId L'identifiant de la tablature
17      * @param $InTabTitle Le titre de la tablature
18      * @param $InTabPath Le chemin de la tablature au format XML
19      * @param $InTabRate La note de la tablature sur 5
20      * @param $InLvlTab Le niveau de la tablature (0 -> facile; 1-> moyen; 2-> difficile)
21      * @param $InUserID L'id de l'utilisateur qui a posté la tablature
22      * @param $InArtistID L'id de l'artiste qui a créé la tablature
23      */
24
25     public function __construct ($InTabId = - 1, $InTabTitle = "", $InTabPath = "", $InTabRate = -1, ←
26     $InLvlTab = -1, $InUserID = - 1,$InArtistID = - 1)
27     {
28         $this->id = $InTabId;

```

```

28     $this->title = $InTabTitle;
29     $this->path = $InTabPath;
30     $this->rate = $InTabRate;
31     $this->lvl = $InLvlTab;
32     $this->userId = $InUserID;
33     $this->artistId = $InArtistID;
34 }
35
36 /**
37  * @brief On ne laisse pas cloner une tablature
38  */
39 private function __clone ()
40 {}
41
42 /**
43  * @brief Est-ce que cet objet est valide
44  *
45  * @return True si valide, autrement false
46  */
47 public function isValid ()
48 {
49     return ( $this->id == -1 || $this->title == "" || $this->path == "" || $this->rate == -1 || ↵
50     $this->lvl == -1 || $this->userId == -1 || $this->artistId == -1) ? false : true;
51 }
52
53 /**
54  * @brief Getter
55  *
56  * @return L'identifiant de la tablature
57  */
58 public function getId ()
59 {
60     return $this->id;
61 }
62
63 /**
64  * @brief Getter
65  *
66  * @return Le titre de la tablature
67  */
68 public function getTitle ()
69 {
70     return $this->title;
71 }
72
73 /**
74  * @brief Getter
75  *
76  * @return Le chemin de la tablature
77  */
78 public function getPath ()
79 {
80     return $this->path;
81 }
82
83 /**
84  * @brief Getter
85  *
86  * @return La note de la tablature
87  */
88 public function getRate ()
89 {
90     return $this->rate;
91 }
92
93 /**
94  * @brief Getter
95  *
96  * @return Le niveau de la tablature
97  */
98 public function getLvl ()
99 {
100     return $this->lvl;
101 }
102
103 /**
104  * @brief Getter
105  *
106  * @return L'identifiant de l'utilisateur qui a posté la tablature
107  */
108 public function getUserId ()
109 {
110     return $this->userId;
111 }
112
113 /**
114  * @brief Getter
115  *
116  * @return L'identifiant de l'artiste qui a créé la tablature
117  */
118 public function getArtistId ()
119 {
120     return $this->artistId;

```

```

121
122 /**
123  * S rialize l'objet en json
124  * @return array|mixed
125  */
126 public function jsonSerialize()
127 {
128     return get_object_vars($this);
129 }
130
131
132 /** Identifiant de la tablature */
133 private $id;
134 /** Le titre de la tablature */
135 private $title;
136 /** Le chemin de la tablature au format XML */
137 private $path;
138 /** La note de la tablature sur 5 */
139 private $rate;
140 /** Le niveau de la tablature (0 -> facile; 1-> moyen; 2-> difficile)*/
141 private $lvl;
142 /** L'id de l'utilisateur qui a post  la tablature */
143 private $userId;
144 /** L'id de l'artiste qui a cr   la tablature */
145 private $artistId;
146 }

```

1.3.8 user.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 15.05.2018
5  * projet: SimpleTab
6  */
7
8 /**
9  * Classe des utilisateurs
10  * @author RS
11  */
12 class Users implements JsonSerializable
13 {
14     /**
15      * @brief Class Constructor
16      * @param $InUserId L'identifiant de l'utilisateur
17      * @param $InUserName Le nom de l'utilisateur
18      * @param $InUserForename Le pr nom de l'utilisateur
19      * @param $InUserPassword Le mot de passe de l'utilisateur
20      * @param $InUserEmail L'email de l'utilisateur
21      * @param $InUserPseudo Le pseudonyme de l'utilisateur
22      */
23     public function __construct ($InUserId = - 1, $InUserName = "", $InUserForename = "", $InUserPassword = ←
24         "", $InUserEmail = "", $InUserPseudo = "")
25     {
26         $this->id= $InUserId;
27         $this->name = $InUserName;
28         $this->forename = $InUserForename;
29         $this->password = $InUserPassword;
30         $this->email = $InUserEmail;
31         $this->pseudo = $InUserPseudo;
32     }
33
34     /**
35      * @brief On ne laisse pas cloner un utilisateur
36      */
37     private function __clone ()
38     {}
39
40     /**
41      * @brief Est-ce que cet objet est valide
42      *
43      * @return True si valide, autrement false
44      */
45     public function isValid ()
46     {
47         return ( $this->id == -1 || $this->name == "" || $this->forename == "" || $this->password == "" || ←
48             $this->email == "" || $this->pseudo == "" ) ? false : true;
49     }
50
51     /**
52      * @brief Getter
53      *
54      * @return L'identifiant de l'utilisateur
55      */
56     public function getId ()
57     {
58         return $this->id;
59     }

```

```

60  /**
61   * @brief Getter
62   *
63   * @return Le nom de l'utilisateur
64   */
65  public function getName ()
66  {
67      return $this->name;
68  }
69
70  /**
71   * @brief Getter
72   *
73   * @return Le prénom de l'utilisateur
74   */
75  public function getForename ()
76  {
77      return $this->forename;
78  }
79
80  /**
81   * @brief Getter
82   *
83   * @return Le mot de passe de l'utilisateur
84   */
85  public function getPassword ()
86  {
87      return $this->password;
88  }
89
90  /**
91   * @brief Getter
92   *
93   * @return L'email de l'utilisateur
94   */
95  public function getEmail()
96  {
97      return $this->email;
98  }
99
100  /**
101   * @brief Getter
102   *
103   * @return Le pseudo de l'utilisateur
104   */
105  public function getPseudo ()
106  {
107      return $this->pseudo;
108  }
109
110  /**
111   * Sériailize l'objet en json
112   * @return array|mixed
113   */
114  public function jsonSerialize()
115  {
116      return get_object_vars($this);
117  }
118
119
120  /** L'identifiant de l'utilisateur */
121  private $id;
122  /** Le nom de l'utilisateur */
123  private $name;
124  /** Le prénom de l'utilisateur */
125  private $forename;
126  /** Le mot de passe de l'utilisateur */
127  private $password;
128  /** L'email de l'utilisateur */
129  private $email;
130  /** Le pseudo de l'utilisateur */
131  private $pseudo;
132
133 }
134 ?>

```

1.3.9 comment.php

```

1  <?php
2  /**
3   * Auteur: romain.ssr@eduge.ch
4   * Date : 17.05.2018
5   * projet: SimpleTab
6   */
7
8  /**
9   * Classe des commentaires
10  * @author RS
11  */
12  class Comment implements JsonSerializable

```

```

13 {
14     /**
15      * @brief Class Constructor
16      * @param $InCommentId L'identifiant du commentaire
17      * @param $InCommentContent Le contenu du commentaire
18      * @param $InTabId L'id de la tablature associée au commentaire
19      * @param $InUserId L'id de l'utilisateur associée au commentaire
20      */
21     public function __construct ($InCommentId = -1, $InCommentContent = "", $InTabId = "", $InUserId = - 1)
22     {
23         $this->id= $InCommentId;
24         $this->commentContent = $InCommentContent;
25         $this->tabId = $InTabId;
26         $this->userId = $InUserId;
27     }
28
29     /**
30      * @brief On ne laisse pas cloner une tablature
31      */
32     private function __clone ()
33     {}
34
35     /**
36      * @brief Est-ce que cet objet est valide
37      *
38      * @return True si valide, autrement false
39      */
40     public function isValid ()
41     {
42         return ( $this->id == -1 || $this->commentContent == "" || $this->tabId == -1 || $this->userId == - 1 ) ? false : true;
43     }
44
45     /**
46      * @brief Getter
47      *
48      * @return L'identifiant du commentaire
49      */
50     public function getId ()
51     {
52         return $this->id;
53     }
54
55     /**
56      * @brief Getter
57      *
58      * @return Le contenu du commentaire
59      */
60     public function getContent ()
61     {
62         return $this->commentContent;
63     }
64
65     /**
66      * @brief Getter
67      *
68      * @return L'id de la tablature associée au commentaire
69      */
70     public function getTabId ()
71     {
72         return $this->tabId;
73     }
74
75     /**
76      * @brief Getter
77      *
78      * @return L'id de l'utilisateur associée au commentaire
79      */
80     public function getUserId ()
81     {
82         return $this->userId;
83     }
84
85     /**
86      * Serialize l'objet en json
87      * @return array|mixed
88      */
89     public function jsonSerialize()
90     {
91         return get_object_vars($this);
92     }
93
94     /** Identifiant du commentaire */
95     private $id;
96     /** Le contenu du commentaire */
97     private $commentContent;
98     /** L'id de la tablature associée au commentaire */
99     private $tabId;
100     /** L'id de l'utilisateur associé au commentaire */
101     private $userId;
102
103 }
104 }

```

1.3.10 Artist.php

```
1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 18.05.2018
5  * projet: SimpleTab
6  */
7
8
9 /**
10 * Classe des artistes
11 * @author RS
12 */
13 class Artist implements JsonSerializable
14 {
15     /**
16      * Artist constructor.
17      * @param int $InArtistId -> L'id de l'artiste
18      * @param string $InArtistName -> Le nom de l'artiste
19      */
20     public function __construct ($InArtistId = - 1, $InArtistName = "")
21     {
22         $this->id= $InArtistId;
23         $this->name = $InArtistName;
24     }
25
26     /**
27      * @brief On ne laisse pas cloner une tablature
28      */
29     private function __clone ()
30     {}
31
32     /**
33      * @brief Est-ce que cet objet est valide
34      *
35      * @return True si valide, autrement false
36      */
37     public function isValid ()
38     {
39         return ( $this->id == -1 || $this->name == "" ) ? false : true;
40     }
41
42     /**
43      * @brief Getter
44      *
45      * @return L'identifiant de l'artiste
46      */
47     public function getId ()
48     {
49         return $this->id;
50     }
51
52     /**
53      * @brief Getter
54      *
55      * @return Le nom de l'artiste
56      */
57     public function getName ()
58     {
59         return $this->name;
60     }
61
62
63     public function jsonSerialize()
64     {
65         return get_object_vars($this);
66     }
67
68     /** Identifiant de l'artiste */
69     private $id;
70     /** Le nom de l'artiste */
71     private $name;
72 }
73 }
```

1.3.11 database.php

```
1 <?php
2 /**
3  * @author dominique.aigroz@edu.ge.ch
4  */
5 require_once '../config/conparam.php';
6
7 /**
8  * @brief Helper class encapsulating
9  *        the PDO object
10 * @author dominique.aigroz@kadeo.net
11 * @remark
```



```

12 */
13
14 class Database
15 {
16     private static $pdoInstance;
17     /**
18      * @brief Class Constructor - Create a new database connection if one doesn't exist
19      * @brief Set to private so no-one can create a new instance via ' = new KDatabase();'
20      * */
21     private function __construct() {}
22     /**
23      * @brief Like the constructor, we make __clone private so nobody can clone the instance
24      * */
25     private function __clone() {}
26     /**
27      * @brief Returns DB instance or create initial connection
28      * @return $objInstance;
29      */
30     public static function getInstance() {
31         if(!self::$pdoInstance){
32             try{
33
34                 $dsn = EDB_DBTYPE.':host='.EDB_HOST.':port='.EDB_PORT.':dbname='.EDB_DBNAME;
35                 self::$pdoInstance = new PDO($dsn, EDB_USER, EDB_PASS, array('charset'=>'utf8'));
36                 self::$pdoInstance->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
37             }catch(PDOException $e){
38                 echo "KDatabase Error: ".$e->getMessage();
39             }
40         }
41         return self::$pdoInstance;
42     } # end method
43     /**
44      * @brief Passes on any static calls to this class onto the singleton PDO instance
45      * @param $chrMethod The method to call
46      * @param $arrArguments The method's parameters
47      * @return $mix The method's return value
48      */
49     final public static function __callStatic( $chrMethod, $arrArguments ) {
50         $pdo = self::getInstance();
51         return call_user_func_array(array($pdo, $chrMethod), $arrArguments);
52     } # end method
53 }

```

1.4 public

1.4.1 css

style.css

```

1 /**
2  * Auteur: romain.ssr@eduge.ch
3  * Date : 18.05.2018
4  * projet: SimpleTab
5  * description: feuille de style
6  */
7 .artistName{
8 color : darkred;
9 }
10 a:hover {
11     text-decoration: underline;
12     cursor: pointer;
13 }

```

1.4.2 js

function.js

```

1 /**
2  * @copyright dominique.aigroz@edu.ge.ch dario.gng@eduge.ch romain.ssr@eduge.ch
3  * Effectue un appel ajax et exécute une fonction en cas de succès
4  * @param string url Le chemin vers le fichier qui va récupérer nos données
5  * @param string callback Le nom de la fonction à exécuter après avoir reçu les données. Il ne faut pas ←
6  * mettre de parenthèses
7  * @param object params Object qui contient tout les paramètres à envoyer
8  * @param boolean async Pour activer / désactiver l'asynchrone d'un appel ajax
9  */
10 function get_data(url, callback, params = {}, async) {
11     // Copier l'objet params dans une variable locale
12     // qui sera simplement utilisées pour l'appel Ajax
13     var dp = params;

```

```

14 // On utilise le paramètre de la fonction qui est stocké sur le stack
15 // pour créer un tableau qui contient les paramètres additionnels qui se
16 // trouvent après params.
17 // Si on stocke ces paramètres dans un tableau créé dans cette fonction,
18 // il sera détruit après l'appel Ajax et on aura plus rien lorsqu'on sera
19 // rappelé de manière asynchrone.
20 params = Array();
21 for (var i = 4; i < arguments.length; i++){
22     params.push(arguments[i]);
23 }
24 $.ajax({
25     method: 'POST',
26     url: url,
27     data: dp,
28     dataType: 'json',
29     async : async,
30     success: function (data) {
31         var msg = '';
32
33         switch (data.ReturnCode){
34             case 0 : // tout bon
35                 // On récupère par params, notre tableau des arguments.
36                 params.unshift(data.Data);
37                 callback.apply(this, params);
38                 break;
39             case 1: // erreur param
40                 alert("Veuillez remplir tous les champs correctement");
41                 break;
42             case 2 : // problème récup données
43                 alert("Un problème est survenu");
44             case 3 : // problème encodage sur serveur
45                 alert("Un problème est survenu");
46
47             default:
48
49                 break;
50         }
51     },
52     error: function(jqXHR){
53         // Votre gestion de l'erreur
54     }
55 });
56 }
57
58 /**
59  * Retourne le niveau de la tablature en toute lettres : 0 -> facile ; 1 -> moyen ; 2 -> difficile
60  * @param $lvlNumber -> le niveau de la tablature en chiffre
61  */
62 function getDifficultyInLetters($lvlNumber)
63 {
64     switch ($lvlNumber)
65     {
66         case "0":
67             return "Facile";
68             break;
69         case "1":
70             return "Moyen";
71             break;
72
73         case "2":
74             return "Difficile";
75             break;
76     }
77 }
78
79 /**
80  * Envoie l'id d'une tablature à la fonction getInfoTab
81  * @param idTab
82  */
83 function passTabIdToGetInfosTab(idTab)
84 {
85     get_data("../controller/getInfosTab.php",getInfosTab,{ 'idTab' : idTab},true);
86     function getInfosTab(data)
87     {
88         if(data != false)
89         {
90             if($.isEmptyObject(data.metadata.capo))
91             {
92                 data.metadata.capo = "";
93             }
94             if($.isEmptyObject(data.metadata.key))
95             {
96                 data.metadata.key = "";
97             }
98             lvlNumber = getDifficultyInNumber(data.metadata.lvl);
99             $('#modifyTitle').val(data.metadata.title);
100             $('#modifyAuthor').val(data.metadata.author);
101             $('#modifyLvl select').val(data.metadata.level);
102             $('#modifyCapo').val(data.metadata.capo);
103             $('#modifyKey').val(data.metadata.key);
104             $('#modifyTuning').val(data.metadata.tuning);
105             $('#modifyTablatureBody').val(data.corpse);
106         }
107     }

```

```

108     }
109 }
110 }
111
112 /**
113  * Supprime la tablature par son id et rafraichit le tableau des tablatures de l'utilisateur
114  * @param idTab
115  * @param idUser
116  */
117 function deleteTabById(idTab, idUser)
118 {
119     if(confirm("Voulez-vous vraiment supprimer la tablature ?")) {
120         get_data("../controller/deleteTabById.php", deleteTabById, {'idTab': idTab}, true);
121
122         function deleteTabById(data) {
123             var message = "";
124             if (data == true) {
125                 message = "<div class=\"alert alert-success text-center\" role=\"alert\">\" +
126                     " Votre tablature a bien été supprimée.</div>";
127             }
128             else {
129                 message = "<div class=\"alert alert-danger text-center\" role=\"alert\">\" +
130                     "Un problème est survenu </div>";
131             }
132             $('#message').append(message);
133             alert('la tablature a été bien supprimée');
134             reloadTabByUSers(idUser);
135         }
136     }
137 }
138 }
139
140 /**
141  * Quand l'administrateur refuse une tablature, elle est supprimée ainsi que la tablature au format XML
142  * @param idTab
143  */
144 function refuseTab(idTab)
145 {
146     get_data("../controller/deleteTabById.php", deleteTabById, {'idTab' : idTab}, true);
147     function deleteTabById(data)
148     {
149         var message = "";
150         if (data == true) {
151             alert('la tablature a été refusée');
152             getAllNonApprovedTabs();
153         }
154         else {
155             alert('un problème est survenu');
156         }
157     }
158 }
159 }
160 }
161 }
162
163 /**
164  * Quand l'administrateur accepte une tablature, son champ approved passe à 1. La tablature devient ↵
165     consultable par tous.
166  * @param idTab
167  */
168 function accepTab(idTab)
169 {
170     get_data("../controller/approuveTab.php", approuveTab, {'idTab' : idTab}, true);
171     function approuveTab(data)
172     {
173         var message = "";
174         if (data == true)
175         {
176             alert('la tablature a été approuvée');
177             getAllNonApprovedTabs();
178         }
179         else
180         {
181             alert('un problème est survenu');
182         }
183     }
184 }
185 }
186
187 /**
188  * rafraichit le tableau des tablatures postées par l'utilisateurs par leur ID
189  * @param idUser
190  */
191 function reloadTabByUSers(idUser)
192 {
193     ↵
194     get_data("../controller/getTabAndRelatedArtistPostedByUser.php", getTabAndRelatedArtistPostedByUser, {'idUser' ↵
195         : idUser}, true);
196     function getTabAndRelatedArtistPostedByUser(data) {
197         $('#tabs').empty();
198         data.forEach(function (tablature) {
199             var artist = $('<a class="artistName" >' + tablature.nameArtist + '</a>');
200             var td = $('<td>').append(artist);

```

```

199     var lvl = getDifficultyInLetters(tablature.lvlTab);
200     var tr = $("|  |
| --- |
|").append(td);
201     $(tr).append(
202         '<td>' + tablature.titleTab + '</td>' +
203         '<td>' + lvl + '</td>' +
204         '<td>' + tablature.rateTab + '</td>' +
205         '<td style="width:1%;><button class="btn btn-dark border-0" style="background-color: ↵
#20262b;" onclick="deleteTabById(' + tablature.idTab + ', ' + idUser + ')"><i class="fas ↵
fa-trash-alt"></i></button></td>' +
206         '<td style="width:1%;><button class="btn btn-dark border-0" data-toggle="modal" ↵
data-target="#modifyTab" style="background-color: #20262b;" onclick="passTabIdToGetInfosTab(' + ↵
tablature.idTab + ')"><i class="fas fa-pencil-alt"></i></button></td>' +
207         '</tr>');
208     $('#tabs').append(tr);
209
210
211     $(artist).click(function () {
212         var artistName = $(this).text();
213         get_data("../controller/getTabByArtist.php", getTabByArtist, {'artistName': artistName}, ↵
true);
214
215         function getTabByArtist(data) {
216             $('#tabs').empty();
217             data.forEach(function (tablature) {
218                 var $lvl = getDifficultyInLetters(tablature.lvlTab);
219                 var $row = $('<tr>' +
220                     '<td><a class="artistName" >' + tablature.nameArtist + '</a></td>' +
221                     '<td>' + tablature.titleTab + '</td>' +
222                     '<td>' + $lvl + '</td>' +
223                     '<td>' + tablature.rateTab + '</td>' +
224                     '</tr>');
225                 $('#tabs').append($row);
226             });
227         }
228     });
229 });
230
231
232 });
233 }
234 }
235
236 /**
237  * Récupère toutes les tablatures non approuvée (champs approved = 0)
238  */
239 function getAllNonApprovedTabs()
240 {
241     get_data("../controller/getAllNonApprovedTab.php", getAllNonApprovedTab, {}, true);
242     function getAllNonApprovedTab(data) {
243         $('#tabs').empty();
244         data.forEach(function (tablature) {
245             var artist = tablature.nameArtist;
246             var td = $(" ").append(artist); 247             var lvl = getDifficultyInLetters(tablature.lvlTab); 248             var tr = $("|").append(td); 249             $(tr).append( 250                 '<td><a class="titleTab">' + tablature.titleTab + '</a></td>' + 251                 '<td>' + lvl + '</td>' + 252                 '<td>' + tablature.rateTab + '</td>' + 253                 '<td style="width:1%;><button class="btn btn-dark border-0" style="background-color: ↵ #20262b;" onclick="accepTab(' + tablature.idTab + ')"><i class="fas fa-check"></i></button></td>' + 254                 '<td style="width:1%;><button class="btn btn-dark border-0" style="background-color: ↵ #20262b;" onclick="refuseTab(' + tablature.idTab + ')"><i class="fas fa-times"></i></button></td>' + 255                 '</tr>'); 256             $('#tabs').append(tr); 257 258 259             $('<.titleTab>').click(function () { 260                 var titleTab = $(this).text(); 261                 get_data("../controller/getTabByTitle.php", getTabByTitle, {'titleTab': titleTab}, true); 262                 function getTabByTitle(data) { 263                     data.forEach(function (tablature) { 264                         window.location.href = "../view/tablaturePage.php?idTab=" + tablature.idTab; 265                     }); 266                 } 267             }); 268         }); 269     } 270 } 271 272 /** 273  * Récupère les utilisateurs et le nombre de tablatures qu'ils ont posté 274  */ 275 function getUsersAndNbTabPosted() 276 { 277     get_data("../controller/getUsersAndNbTabPosted.php", getUsersAndNbTabPosted, {}, true); 278     function getUsersAndNbTabPosted(data) { 279         $('#users').empty(); 280         data.forEach(function (user) { 281             var pseudo = user.pseudoUser; 282             var email = user.emailUser; 283             var nbTab = user.nbTab; 284             var tr = $("|"); 285             $(tr).append(  |  |  |  | |

```

```

286         '<td>' + pseudo + '</td>' +
287         '<td>' + email + '</td>' +
288         '<td>' + nbTab + '</td>' +
289         '<td style="width:1%;"><button class="btn btn-dark border-0" style="background-color: ↵
#20262b;" onclick="deleteUserById(' + user.idUsers + ')"><i class="fas ↵
fa-trash-alt"></i></button></td>' +
        '</tr>');
290     $('#users').append(tr);
291 });
292 }
293 }
294 }
295
296 /**
297  * Supprime la tablatures à partir d'un ID
298  * @param idUser
299  */
300 function deleteUserById(idUser)
301 {
302     if(confirm("Voulez-vous vraiment supprimer l'utilisateur ?")) {
303         get_data("../controller/deleteUserById.php", deleteUserById, {'idUser': idUser}, true);
304
305         function deleteUserById(data) {
306             var message = "";
307             if (data == true) {
308                 alert("l'utilisateur a été supprimé avec succès");
309                 getAllNonApprouvedTabs();
310                 getUsersAndNbTabPosted();
311             }
312             else {
313                 alert('un problème est survenu');
314             }
315         }
316     }
317 }
318 }
319 }
320 }
321
322 function updateTabRate()
323 {
324     get_data("../controller/deleteUserById.php",deleteUserById,{'idUser' : idUser},true);
325 }
326 }
327
328 /**
329  * Renvoie la difficulté d'une tablature en toute lettres depuis un chiffre
330  * @param $lvlInLetter
331  * @returns le chiffre correspondant à la difficulté
332  */
333 function getDifficultyInNumber($lvlInLetter)
334 {
335     $lvl = $.trim($lvlInLetter);
336     switch ($lvl)
337     {
338         case "Facile":
339             return 0;
340             break;
341         case "Moyen":
342             return 1;
343             break;
344         case "Difficile":
345             return 2;
346             break;
347     }
348 }
349 }
350
351 function identifyUser(pseudoOrMail,password)
352 {
353
354     get_data("../controller/identifyUser.php",identifyUser,{'mailOrPseudo' :pseudoOrMail, 'pwdConnexion' : ↵
password},true);
355     function identifyUser(data) {
356         if(data.length != 0)
357         {
358             data.forEach(function(user) {
359                 location.reload();
360             });
361         }
362         else
363         {
364             alert("l'utilisateur n'existe pas ou le mot de passe est incorrect")
365             get_data("../controller/destroySession.php",destroySession,{},true);
366             function destroySession(data)
367             {
368             }
369         }
370     }
371 }
372 }
373 }

```

1.5 tabs

1.5.1 81.php

```
1 <?php
2 $xmlstr = <<<XML
3 <?xml version = "1.0" encoding="UTF-8" standalone="yes" ?>
4 <tabs>
5     <metadata>
6         <title> Hallelujah </title>
7         <author>Jeff Buckley</author>
8         <tuning>E A D G B E</tuning>
9         <capo></capo>
10        <key></key>
11        <level> Facile</level>
12    </metadata>
13    <corpse>
14        [Intro]
15    C Am C Am
16
17 [Verse 1]
18 C Am
19 I heard there was a secret chord
20 C Am
21 That David played and it pleased the lord
22 F G C G
23 But you don't really care for music, do you?
24 C F G
25 Well it goes like this the fourth, the fifth
26 Am F
27 The minor fall and the major lift
28 G E7 Am
29 The baffled king composing hallelujah
30
31 [Chorus]
32 F Am F C G C G
33 Hallelujah, hallelujah, hallelujah, hallelu-u-u-jah ....
34
35 [Verse 2]
36 C Am
37 Well your faith was strong but you needed proof
38 C Am
39 You saw her bathing on the roof
40 F G C G
41 Her beauty and the moonlight overthrew you
42 C F G
43 She tied you to her kitchen chair
44 Am F
45 She broke your throne and she cut your hair
46 G E7 Am
47 And from your lips she drew the hallelujah
48
49 [Chorus]
50 F Am F C G C G
51 Hallelujah, hallelujah, hallelujah, hallelu-u-u-jah ....
52
53 [Verse 3]
54 C Am
55 Baby I've been here before
56 C Am
57 I've seen this room and I've walked this floor
58 F G C G
59 I used to live alone before I knew you
60 C F G
61 I've seen your flag on the marble arch
62 Am F
63 But love is not a victory march
64 G E7 Am
65 It's a cold and it's a broken hallelujah
66
67 [Chorus]
68 F Am F C G C G
69 Hallelujah, hallelujah, hallelujah, hallelu-u-u-jah ....
70
71 [Verse 4]
72 C Am
73 Well there was a time when you let me know
74 C Am
75 What's really going on below
76 F G C G
77 But now you never show that to me do you
78 C F G
79 But remember when I moved in you
80 Am F
81 And the holy dove was moving too
```

```

89      G          E7          Am
90 And every breath we drew was hallelujah
91
92
93 [Chorus]
94      F          Am          F          C          G          C          G
95 Hallelujah, hallelujah, hallelujah, hallelu-u-u-u-jah ....
96
97
98 [Verse 5]
99      C          Am
100 Well, maybe there's a god above
101      C          Am
102 But all I've ever learned from love
103      F          G          C          G
104 Was how to shoot somebody who outdrew you
105      C          F          G
106 It's not a cry that you hear at night
107      Am          F
108 It's not somebody who's seen the light
109      G          E7          Am
110 It's a cold and it's a broken hallelujah
111
112
113 [Outro]
114      F          Am          F          C          G          C          G
115 Hallelujah, hallelujah, hallelujah, hallelu-u-u-u-jah ....
116 </corpse>
117 </tabs>
118 XML;
119 ?>

```

1.5.2 83.php

```

1 <?php
2 $xmlstr = <<<XML
3 <?xml version = "1.0" encoding="UTF-8" standalone="yes" ?>
4 <tabs>
5     <metadata>
6         <title> Perfect </title>
7         <author>Ed Sheeran</author>
8         <tuning>E A D G B E</tuning>
9         <capo></capo>
10        <key></key>
11        <level> Facile</level>
12    </metadata>
13    <corpse>
14        Play this song with the regular shapes of the chords given in this song
15    or use the ones that can be heard on the recording:
16
17        E-A-D-G-B-e
18    G      3-x-0-0-3(3)
19    Em     0-2-2-0-3(3) (= Em7)
20    C      x-3-2-0-3(3) (= Cadd9)
21
22    [Intro]
23    G
24
25    [Verse]
26          G          Em
27 I found a love for me
28          C          D
29 Darling just dive right in, and follow my lead
30          G          Em
31 Well I found a girl beautiful and sweet
32          C          D
33 I never knew you were the someone waiting for me
34
35    [Pre-Chorus]
36          G
37 Cause we were just kids when we fell in love
38          Em          C          G D
39 Not knowing what it was, I will not give you up this ti-me
40          G          Em
41 But darling just kiss me slow, your heart is all I own
42          C          D
43 And in your eyes you're holding mine
44
45    [Chorus]
46          Em C          G          D          Em
47 Baby, I'm dancing in the dark, with you between my arms
48          G D          Em
49 Barefoot on the grass, listening to our favorite song
50          C          G          D          Em
51 When you said you looked a mess, I whispered underneath my breath
52          C          G          D          G
53 But you heard it, darling you look perfect tonight
54
55 | G D/F# Em D | C D |
56

```

```

57 [Verse]
58           G           Em
59 Well I found a woman, stronger than anyone I know
60           C           D
61 She shares my dreams, I hope that someday I'll share her home
62           G           Em
63 I found a love, to carry more than just my secrets
64           C           D
65 To carry love, to carry children of our own
66
67 [Pre-Chorus]
68           G           Em
69 We are still kids, but we're so in love, fighting against all odds
70           C           G D
71 I know that we'll be alright this ti-me
72           G           Em
73 Darling just hold my hand, be my girl, I'll be your man
74           C           D
75 I see my future in your eyes
76
77 [Chorus]
78           Em C           G           D           Em
79 Baby, I'm dancing in the dark, with you between my arms
80           C           G D           Em
81 Barefoot on the grass, listening to our favorite song
82           C           G           D
83 When I saw you in that dress, looking so beautiful
84           Em C           G D [ G ]
85 I don't deserve this, darling you look perfect tonight
86
87 [Interlude]
88 | G | % | Em | % |
89 | C | % | D | % |
90
91 [Chorus]
92           Em C           G           D           Em
93 Baby, I'm dancing in the dark, with you between my arms
94           C           G D           Em
95 Barefoot on the grass, listening to our favorite song
96           C           G           D           Em
97 I have faith in what I see, now I know I have met an angel
98           C           G           D
99 In person, and she looks perfect
100
101 [Outro]
102           G/B C           Dadd4 D [ G ]
103 I don't deserve this, you look perfect tonight
104
105 | G D/F# Em D | C D | G
106 </corpse>
107 </tabs>
108 XML;
109 ?>

```

1.5.3 84.php

```

1 <?php
2 $xmlstr = <<<XML
3 <?xml version = "1.0" encoding="UTF-8" standalone="yes" ?>
4 <tabs>
5     <metadata>
6         <title> Photograph </title>
7         <author>Ed Sheeran</author>
8         <tuning>E A D G B E</tuning>
9         <capo></capo>
10        <key></key>
11        <level> Facile</level>
12    </metadata>
13    <corpse>
14        [Intro]
15
16    D   Bm   A   G
17
18
19 [Verse]
20
21           D
22 Loving can hurt
23           Bm
24 Loving can hurt sometimes
25           A           G
26 But it's the only thing that I know
27           D
28 When it gets hard
29           Bm
30 You know it can get hard sometimes
31           A           G
32 It is the only thing that makes us feel alive
33
34

```



```

35 [Pre-Chorus]
36
37 Bm          G
38 We keep this love in a photograph
39 D          A
40 We make these memories for ourselves
41 Bm
42 Where our eyes are never closing
43 G
44 Our hearts were never broken
45 D          A
46 And times forever frozen still
47
48
49 [Chorus]
50
51 D
52 So you can keep me inside the pocket of your
53 A
54 Ripped jeans holding me closer till our
55 Bm          G
56 Eyes meet, you won't ever be alone
57 D
58 Wait for me to come home
59
60
61 [Verse]
62
63 D
64 Loving can heal
65 Bm
66 Loving can mend your soul
67 A          G
68 And it's the only thing that I know
69 D
70 I swear it will get easier
71 Bm
72 remember that with every piece of ya
73 A          G
74 It is the only thing we take with us when we die
75
76
77 [Pre-Chorus]
78
79 Bm          G
80 We keep this love in a photograph
81 D          A
82 We make these memories for ourselves
83 Bm
84 Where our eyes are never closing
85 G
86 Our hearts were never broken
87 D          A
88 And times forever frozen still
89
90
91 [Chorus]
92
93 D
94 So you can keep me inside the pocket of your
95 A
96 Ripped jeans holding me closer till our
97 Bm          G
98 Eyes meet, you won't ever be alone
99
100
101 [Bridge]
102
103 D
104 And if you hurt me that's ok baby, only
105 A
106 Words bleed inside these pages you just
107 Bm          G
108 Hold me and I won't ever let you go
109
110
111 [Interlude]
112
113 Bm
114 Wait for me to come home
115 G
116 Wait for me to come home
117 D
118 Wait for me to come home
119 A
120 Wait for me to come home
121
122
123 [Outro]
124
125 D
126 Oh you can fit me inside the necklace you got when you were
127 A
128 16 next to your heartbeat where I

```

```

129 Bm G
130 Should be, keep it deep within your soul
131 D
132 And if you hurt me that's ok baby only
133 A
134 Words bleed inside these pages you just
135 Bm G
136 Hold me and I won't ever let you go
137 D
138 When I'm away I will remember how you
139 A Bm
140 Kissed me under the lamp post back on sixth street
141 G
142 Hearing you whisper through the phone
143 D
144 Wait for me to come home
145
146
147
148 </corpse>
149 </tabs>
150 XML;
151 ?>

```

1.6 view

1.6.1 homePage.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 09.05.2018
5  * projet: SimpleTab
6  * description : page d'accueil du site
7  */
8
9 session_start();
10
11 require_once "modals.php";
12
13 $navIdOrButton = "";
14 $navMenu1 = "";
15 $navMenu2 = "";
16
17 if(isset($_SESSION['user']))
18 {
19     $navIdOrButton = "<div class=\"dropdown\">
20         <button type=\"button\" onclick=\"\" class=\"btn btn-light dropdown-toggle\" ↵
21         data-toggle=\"dropdown\" aria-haspopup=\"true\" aria-expanded=\"false\"> ↵
22         \"$_SESSION['user'][0]['pseudoUser']\"</button>
23         <div class=\"dropdown-menu\" aria-labelledby=\"dropdownMenuButton\">
24             <a class=\"dropdown-item\" ↵
25             href=\"../controller/destroySession.php\">Déconnexion</a>
26         </div>
27     </div>";
28     if($_SESSION['user'][0]['role_idrole'] == 0)
29     {
30         $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
31         class=\"sr-only\">(current)</span></a></h5>";
32         $navMenu2 = "<h5><a class=\"nav-link\" href=\"../view/tablatreManagerPage.php\">Gestion des ↵
33         tablatures </a></h5>";
34     }
35     elseif($_SESSION['user'][0]['role_idrole'] == 1)
36     {
37         $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
38         class=\"sr-only\">(current)</span></a></h5>";
39         $navMenu2 = "<h5><a class=\"nav-link\" href=\"../view/tablatreAndUserManagerPage.php\">Gestion des ↵
40         tablatures et utilisateurs </a></h5>";
41     }
42 }
43 else
44 {
45     $navIdOrButton = "<button type=\"button\" class=\"btn btn-light\" data-toggle=\"modal\" ↵
46     data-target=\"#addUser\">S'inscrire</button> | <button type=\"button\" class=\"btn ↵
47     btn-light\" data-toggle=\"modal\" data-target=\"#connectUser\">S'identifier</button>";
48     $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
49     class=\"sr-only\">(current)</span></a></h5>";
50 }
51 ?>
52 <!DOCTYPE HTML>
53 <html>
54 <head>
55     <title>Accueil</title>
56     <script src="https://code.jquery.com/jquery-3.3.1.js" ↵
57     integrity="sha256-2Kok7Mb0yxpqUVvAk/HJ2jigOSYS2auK4Pfbzm7uH60=" crossorigin="anonymous"></script>
58     <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" ↵
59     integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/18WvCWPiPm49" ↵
60     crossorigin="anonymous"></script>

```

```

48 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" ↵
    integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQpOMk8ck+5T" ↵
    crossorigin="anonymous"></script>
49 <link rel="stylesheet" type="text/css" href=" ../public/css/style.css">
50 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css" ↵
    integrity="sha384-9gVQ4dYFwwWSjIDZLnLEWnxCjeSWFphJiWGPXr1jddIhOegui1Fw05qRgvFX0dJZ4" ↵
    crossorigin="anonymous">
51
52 <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.0.13/css/all.css" ↵
    integrity="sha384-DNOH268U8hZfKXOortjWvjxusGo9WQnrNx2sqG0tfsghAvtVRW3tvkXWZ58N99jp" ↵
    crossorigin="anonymous">
53 <!-- JQuery CDN -->
54 </head>
55 <body>
56 <div class="border ">
57   <div class=".col text-right mr-3 mb-0"> <?php echo $navIdOrButton;?>
58
59   <div id="body" class=" mx-5">
60     <nav class="navbar navbar-expand-lg navbar-light p-0">
61       <a class="navbar-brand" href=" ../view/homePage.php" id="logo">
62         
63       </a>
64
65       <div class="collapse navbar-collapse" id="navbarSupportedContent">
66         <ul class="navbar-nav mr-auto " style="margin: auto;">
67           <li class="nav-item active">
68             <?php echo $navMenu1?>
69           </li>
70           <li class="nav-item active">
71             <?php echo $navMenu2?>
72           </li>
73         </ul>
74
75         <form class="form-inline my-2 my-lg-0" method="get" action=" ../view/search.php">
76
77           <input class="form-control mr-sm-0" type="search" placeholder="Rechercher" ↵
78             aria-label="Search" id="searchBar" name="nameArtistOrTitleTab">
79           <button type="submit" class=" btn btn-outline-secondary ">
80             <i class="fas fa-search"></i>
81           </button>
82         </form>
83       </div>
84     </nav>
85   </div>
86 </div>
87 </div>
88
89 </div>
90 <div class=" m-5">
91   <table class="table table-dark">
92     <thead style="color:red">
93       <tr>
94         <th scope="col">Artiste</th>
95         <th scope="col">Titre</th>
96         <th scope="col">Difficult </th>
97         <th scope="col">Note</th>
98       </tr>
99     </thead>
100     <tbody id='tabs'>
101     </tbody>
102   </table>
103 </div>
104 </body>
105 <script src=" ../public/js/function.js"></script>
106 <script type="text/javascript">
107   $( document ).ready(function() {
108
109     get_data(" ../controller/getTabAndRelatedArtist.php",getTabAndRelatedArtist,{},true);
110     function getTabAndRelatedArtist(data) {
111       $('#tabs').empty();
112       data.forEach(function(tablature){
113         var artist = $('<a class="artistName" >' + tablature.nameArtist + '</a>');
114         var td = $('<td>').append(artist);
115         var lvl = getDifficultyInLetters(tablature.lvlTab);
116         var tr = $('<tr>').append(td);
117         $(tr).append(
118           '<td><a class="titleTab">' +tablature.titleTab+'</a></td>' +
119           '<td>' + lvl + '</td>' +
120           '<td>' +tablature.rateTab+'</td>' +
121           '</tr>');
122         $('#tabs').append(tr);
123
124
125         $(artist).click(function () {
126           var artistName = $(this).text();
127           ↵
128           get_data(" ../controller/getTabByArtist.php",getTabByArtist,{ 'artistName':artistName},true);
129           function getTabByArtist(data) {
130             $('#tabs').empty();
131             data.forEach(function(tablature){
132               var $lvl = getDifficultyInLetters(tablature.lvlTab);
133               var $row = $('<tr>' +

```

```

134         '<td>'+tablature.titleTab+'</td>' +
135         '<td>'+ $lvl +'</td>' +
136         '<td>'+tablature.rateTab+'</td>' +
137         '</tr>');
138     $('#tabs').append($row);
139
140     });
141 }
142 });
143
144 $('#titleTab').click(function () {
145     var titleTab = $(this).text();
146     get_data("../controller/getTabByTitle.php",getTabByTitle,{ 'titleTab':titleTab},true);
147     function getTabByTitle(data) {
148         data.forEach(function(tablature) {
149             window.location.href= "../view/tablaturePage.php?idTab="+tablature.idTab;
150         });
151     }
152 });
153
154 });
155
156
157 }
158
159
160
161
162 function hideModal() {
163     $('#addUser').modal('hide')
164 }
165
166 $('#btnAddUser').click(function () {
167     var name = $('#name').val();
168     var forename = $('#forename').val();
169     var password = $('#password').val();
170     var passwordConfirm = $('#passwordConfirm').val();
171     var email = $('#email').val();
172     var pseudo = $('#pseudo').val();
173     get_data("../controller/getUserByPseudo.php",getUserByPseudo,{ 'pseudo' : pseudo},true);
174     function getUserByPseudo(dataPseudo)
175     {
176         if(dataPseudo == false)
177         {
178             get_data("../controller/getUserByEmail.php",getUserByEmail,{ 'email' : email},true);
179             function getUserByEmail(dataEmail)
180             {
181                 if(dataEmail == false)
182                 {
183                     get_data("../controller/addUser.php",addUser,{ 'name' : name, 'forename' : ↵
forename, 'password' : password, 'email' : email, 'pseudo' : pseudo, 'passwordConfirm' : ↵
passwordConfirm},true);
184                     function addUser(){
185                         identifyUser(pseudo,password);
186                         $('#addUser').modal('hide');
187                     }
188                 }
189             }
190             else
191             {
192                 alert('Ce mail est déjà pris, veuillez en utiliser un autre');
193             }
194         }
195     }
196 }
197
198 else
199 {
200     alert('Ce pseudo est déjà pris, veuillez en utiliser un autre');
201 }
202 }
203 });
204
205 $('#btnIdentifyUser').click(function () {
206     var mailOrPseudo = $('#pseudoOrMail').val();
207     var pwdConnexion = $('#pwdConnexion').val();
208     identifyUser(mailOrPseudo,pwdConnexion);
209     $('#connectUser').modal('hide');
210 }
211 });
212 });
213 </script>

```

1.6.2 search.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 17.05.2018
5  * projet: SimpleTab

```

```

6  * description : page de résultat des recherches
7  */
8
9
10 session_start();
11
12 require_once "modals.php";
13
14 $navIdOrButton = "";
15 $navMenu1 = "";
16 $navMenu2 = "";
17
18 if(isset($_SESSION['user']))
19 {
20     $navIdOrButton = "<div class=\"dropdown\">
21         <button type=\"button\" onclick=\"\" class=\"btn btn-light dropdown-toggle\" ↵
22         data-toggle=\"dropdown\" aria-haspopup=\"true\" aria-expanded=\"false\"> ↵
23         \"._SESSION['user'][0]['pseudoUser']\"</button>
24         <div class=\"dropdown-menu\" aria-labelledby=\"dropdownMenuButton\">
25             <a class=\"dropdown-item\" ↵
26             href=\"../controller/destroySession.php\">Déconnexion</a>
27         </div>
28     </div>";
29     if($_SESSION['user'][0]['role_idrole'] == 0)
30     {
31         $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
32         class=\"sr-only\">(current)</span></a></h5>";
33         $navMenu2 = "<h5><a class=\"nav-link\" href=\"../view/tablatureManagerPage.php\">Gestion des ↵
34         tablatures </a></h5>";
35     }
36     elseif($_SESSION['user'][0]['role_idrole'] == 1)
37     {
38         $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
39         class=\"sr-only\">(current)</span></a></h5>";
40         $navMenu2 = "<h5><a class=\"nav-link\" href=\"../view/tablatureAndUserManagerPage.php\">Gestion des ↵
41         tablatures et utilisateurs </a></h5>";
42     }
43 }
44 else
45 {
46     $navIdOrButton = "<button type=\"button\" class=\"btn btn-light\" data-toggle=\"modal\" ↵
47     data-target=\"#addUser\">S'inscrire</button> | <button type=\"button\" class=\"btn ↵
48     btn-light\" data-toggle=\"modal\" data-target=\"#connectUser\">S'identifier</button>";
49     $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
50     class=\"sr-only\">(current)</span></a></h5>";
51 }
52 ?>
53 <!DOCTYPE HTML>
54 <html>
55 <head>
56 <title>Recherche</title>
57 <script src="https://code.jquery.com/jquery-3.3.1.js" ↵
58 integrity="sha256-2Kok7Mb0yxpGUvAk/HJ2jigOSYS2auK4PfzBm7uH60=" crossorigin="anonymous"></script>
59 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.14.3/umd/popper.min.js" ↵
60 integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPiPM49" ↵
61 crossorigin="anonymous"></script>
62 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" ↵
63 integrity="sha384-smHYKdLADwkX01EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQp0Mk8ck+5T" ↵
64 crossorigin="anonymous"></script>
65 <link rel="stylesheet" type="text/css" href="../public/css/style.css">
66 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css" ↵
67 integrity="sha384-9gVQ4dYFwwWSjIDZnLEWnxCjeSWFphJiWGPXr1jddIhOegiu1Fw05qRGvFX0dJZ4" ↵
68 crossorigin="anonymous">
69
70 <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.0.13/css/all.css" ↵
71 integrity="sha384-DNOHZ68U8hZfKX0rtjWvjxusGo9WQnrNx2sqG0tfsghAvtVRW3tvkXWZh58N9jp" ↵
72 crossorigin="anonymous">
73 <!-- JQuery CDN -->
74 </head>
75 <body>
76 <div class="border">
77     <div class="col text-right mr-3 mb-0"> <?php echo $navIdOrButton;?>
78
79     <div id="body" class="mx-5">
80         <nav class="navbar navbar-expand-lg navbar-light p-0">
81             <a class="navbar-brand" href="../view/homePage.php" id="logo">
82                 
83             </a>
84
85             <div class="collapse navbar-collapse" id="navbarSupportedContent">
86                 <ul class="navbar-nav mr-auto" style="margin: auto;">
87                     <li class="nav-item active">
88                         <?php echo $navMenu1?>
89                     </li>
90                     <li class="nav-item active">
91                         <?php echo $navMenu2?>
92                     </li>
93                 </ul>
94
95                 <form class="form-inline my-2 my-lg-0">
96                     <input class="form-control mr-sm-0" type="search" placeholder="Rechercher" ↵
97                     aria-label="Search" id="searchBar">
98                 </form>
99             </div>
100         </nav>
101     </div>
102 </div>

```

```

79         <button type="button" class=" btn btn-outline-secondary " ↵
onclick="search($('#searchBar').val())">
80             <i class="fas fa-search"></i>
81         </button>
82
83     </form>
84 </div>
85 </nav>
86 </div>
87 </div>
88 </div>
89
90 </div>
91 <div class=" m-5">
92     <table class="table table-dark">
93         <thead style="color:red">
94             <tr>
95                 <th scope="col">Artiste</th>
96                 <th scope="col">Titre</th>
97                 <th scope="col">Difficult </th>
98                 <th scope="col">Note</th>
99             </tr>
100         </thead>
101         <tbody id='tabs'>
102             </tbody>
103     </table>
104 </div>
105 </body>
106 <script src="../../public/js/function.js"></script>
107 <script type="text/javascript">
108     $( document ).ready(function() {
109         ↵
110         var tabTitleOrArtistName = <?php echo ''; echo $_GET['nameArtistOrTitleTab']; echo '';?>;
111         get_data("../../controller/getTabAndRelatedArtistByName.php",getTabAndRelatedArtistByName,{ 'tabTitleOrArtistName' ↵
:tabTitleOrArtistName},true);
112         function getTabAndRelatedArtistByName(data)
113         {
114             $('#tabs').empty();
115             data.forEach(function(tablature){
116                 var artist = $('<a class="artistName" >' + tablature.nameArtist + '</a>');
117                 var td = $('<td>').append(artist);
118                 var lvl = getDifficultyInLetters(tablature.lvlTab);
119                 var tr = $('<tr>').append(td);
120                 $(tr).append(
121                     '<td>' + tablature.titleTab + '</td>' +
122                     '<td>' + lvl + '</td>' +
123                     '<td>' + tablature.rateTab + '</td>' +
124                     '</tr>');
125                 $('#tabs').append(tr);
126
127                 $(artist).click(function () {
128                     var artistName = $(this).text();
129                     ↵
130                     get_data("../../controller/getTabByArtist.php",getTabByArtist,{ 'artistName':artistName},true);
131                     function getTabByArtist(data) {
132                         $('#tabs').empty();
133                         data.forEach(function(tablature){
134                             var $lvl = getDifficultyInLetters(tablature.lvlTab);
135                             var $row = $('<tr>' +
136                                 '<td><a class="artistName" >' + tablature.nameArtist + '</a></td>' +
137                                 '<td>' + tablature.titleTab + '</td>' +
138                                 '<td>' + $lvl + '</td>' +
139                                 '<td>' + tablature.rateTab + '</td>' +
140                                 '</tr>');
141                             $('#tabs').append($row);
142                         });
143                     }
144                 });
145             });
146         }
147
148         $('#btnAddUser').click(function () {
149             var name = $('#name').val();
150             var forename = $('#forename').val();
151             var password = $('#password').val();
152             var passwordConfirm = $('#passwordConfirm').val();
153             var email = $('#email').val();
154             var pseudo = $('#pseudo').val();
155             get_data("../../controller/getUserByPseudo.php",getUserByPseudo,{ 'pseudo' : pseudo},true);
156             function getUserByPseudo(dataPseudo)
157             {
158                 if(dataPseudo == false)
159                 {
160                     get_data("../../controller/getUserByEmail.php",getUserByEmail,{ 'email' : email},true);
161                     function getUserByEmail(dataEmail)
162                     {
163                         if(dataEmail == false)
164                         {
165                             get_data("../../controller/addUser.php",addUser,{ 'name' : name, 'forename' : ↵
forename, 'password' : password, 'email' : email, 'pseudo' : pseudo, 'passwordConfirm' : ↵
passwordConfirm},true);
166                             function addUser(){

```

```

167         identifyUser(pseudo,password);
168         $('#addUser').modal('hide');
169     }
170 }
171 }
172 else
173 {
174     alert('Ce mail est déjà pris, veuillez en utiliser un autre');
175 }
176 }
177 }
178 }
179 }
180 else
181 {
182     alert('Ce pseudo est déjà pris, veuillez en utiliser un autre');
183 }
184 }
185 });
186
187 $('#btnIdentifyUser').click(function () {
188     var mailOrPseudo = $('#pseudoOrMail').val();
189     var pwdConnexion = $('#pwdConnexion').val();
190     identifyUser(mailOrPseudo,pwdConnexion);
191     $('#connectUser').modal('hide');
192 }
193 });
194 });
195 </script>

```

1.6.3 tablaturePage.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 17.05.2018
5  * projet: SimpleTab
6  * description : page des détails d'une tablature
7  */
8
9 session_start();
10
11 require_once "../view/modals.php";
12
13 if(isset($_SESSION['user']))
14 {
15     $userId = $_SESSION['user'][0]['idUsers'];
16     $pseudoUser = $_SESSION['user'][0]['pseudoUser'];
17 }
18
19 if(isset($_GET['idTab']))
20 {
21     $idTab = $_GET['idTab'];
22     include "../tabs/".$idTab.".php";
23     $tabs = new SimpleXMLElement($xmlstr);
24
25     $idTab = $_GET['idTab'];
26     $title = $tabs->metadata->title;
27     $nameArtist = $tabs->metadata->author;
28     $tuning = $tabs->metadata->tuning;
29     $capo = $tabs->metadata->capo;
30     $key = $tabs->metadata->key;
31     $lvl = $tabs->metadata->level;
32     $bodyTab = $tabs->corpse;
33
34     $body = "<div id='tab' class='p-3 border' style='display: table; margin:0 auto;'>
35     <div id='metadatas' >
36         <table>
37             <tr>
38                 <td> Titre :</td>
39                 <td>$title </td>
40             </tr>
41             <tr>
42                 <td> Auteur :</td>
43                 <td> $nameArtist</td>
44             </tr>
45             <tr>
46                 <td> Accordage :</td>
47                 <td> $tuning</td>
48             </tr>
49             <tr>
50                 <td> Capo :</td>
51                 <td>$capo </td>
52             </tr>
53             <tr>
54                 <td> Tonalité :</td>
55                 <td> $key</td>
56             </tr>
57             <tr>
58                 <td> Difficulté :</td>

```



```

59         <td>$lvl </td>
60     </tr>
61 </table>
62 </div>
63 <div id='tabBody' >
64     <table>
65         <tr>
66             <td>
67                 <pre>$bodyTab </pre>
68             </td>
69         </tr>
70     </table>
71 </div>
72 <div class='border-top' id='commentSection'>
73     <table class='col'>
74 <tr>
75         <td colspan='2' id='Score' class='text-right'><i id='1' class='far fa-star rateStar'></i><i id='2' class='far fa-star rateStar'></i><i id='3' class='far fa-star rateStar'></i><i id='4' class='far fa-star rateStar'></i><i id='5' class='far fa-star rateStar'></i></td>
76     </tr>
77         <td colspan='2' id="averageScore" class='text-right'> note moyenne : </td>
78     </tr>
79     <tr id='comments'>
80         <td colspan='2'>Commentaire :</td>
81     </tr>
82     <tr id='addComment'>
83         <td colspan='2'><textarea style='width: 100%;' id='myComment' placeholder='Que pensez-vous de cette tablature ?'></textarea><button style='float: right; display: inline;' id='postComment'>Soumettre</button></td>
84     </tr>
85 </table>
86 </div>
87 </div>
88 </div>
89
90 ";
91
92
93
94 }
95 }
96 else
97 {
98     $body = "<div class=\"alert alert-danger text-center\" role=\"alert\">
99         Aucune Tablature sélectionnée, veuillez retourner à <a class=\"alert-link\" href='../view/homePage.php'>la page d'Accueil</a> et cliquez sur le titre pour sélectionner une tablature.
100     </div>";
101 }
102
103 $navIdOrButton="";
104 $navMenu1 = "";
105 $navMenu2 = "";
106
107 if(isset($_SESSION['user']))
108 {
109     $navIdOrButton = "<div class=\"dropdown\">
110         <button type=\"button\" onclick=\"\" class=\"btn btn-light dropdown-toggle\" data-toggle=\"dropdown\" aria-haspopup=\"true\" aria-expanded=\"false\">
111             \"$_SESSION['user'][0]['pseudoUser']\"</button>
112         <div class=\"dropdown-menu\" aria-labelledby=\"dropdownMenuButton\">
113             <a class=\"dropdown-item\" href=\"../controller/destroySession.php\">Déconnexion</a>
114         </div>
115     </div>";
116     if($_SESSION['user'][0]['role_idrole'] == 0)
117     {
118         $navMenu1 = "<h5><a class=\"nav-link\" href='../view/homePage.php'>Accueil &ltspan class=\"sr-only\">(current)</span></a></h5>";
119         $navMenu2 = "<h5><a class=\"nav-link\" href='../view/tablatureManagerPage.php'>Gestion des tablatures </a></h5>";
120     }
121     elseif($_SESSION['user'][0]['role_idrole'] == 1)
122     {
123         $navMenu1 = "<h5><a class=\"nav-link\" href='../view/homePage.php'>Accueil &ltspan class=\"sr-only\">(current)</span></a></h5>";
124         $navMenu2 = "<h5><a class=\"nav-link\" href='../view/tablatureAndUserManagerPage.php'>Gestion des tablatures et utilisateurs </a></h5>";
125     }
126 }
127 else
128 {
129     $navIdOrButton = "<button type=\"button\" class=\"btn btn-light\" data-toggle=\"modal\" data-target=\"#addUser\">S'inscrire</button> | <button type=\"button\" class=\"btn btn-light\" data-toggle=\"modal\" data-target=\"#connectUser\">S'identifier</button>";
130     $navMenu1 = "<h5><a class=\"nav-link\" href='../view/homePage.php'>Accueil &ltspan class=\"sr-only\">(current)</span></a></h5>";
131 }
132 }
133
134 <!DOCTYPE HTML>
135 <html>
136 <head>

```



```

137 <title>Page d'une tablature</title>
138 <script src="https://code.jquery.com/jquery-3.3.1.js" ↵
139 integrity="sha256-2Kok7Mb0yxpqUVvAk/HJ2jigOSYS2auK4Pfzbm7uH60=" crossorigin="anonymous"></script>
140 <script src="https://cdn.jsdelivr.net/npm/popper.js/1.14.3/umd/popper.min.js" ↵
141 integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPiPM49" ↵
142 crossorigin="anonymous"></script>
143 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" ↵
144 integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQpOMk8ck+5T" ↵
145 crossorigin="anonymous"></script>
146 <link rel="stylesheet" type="text/css" href="../public/css/style.css">
147 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css" ↵
148 integrity="sha384-9gVQ4dYFwwWSjIDZnLEWnxCjeSfWphJiwGPXr1jddIh0egiu1Fw05qRGrvFX0dJZ4" ↵
149 crossorigin="anonymous">
150 <!-- JQuery CDN -->
151 </head>
152 <body>
153 <div class="border ">
154 <div class=".col text-right mr-3 mb-0"> <?php echo $navIdOrButton;?>
155
156 <div id="body" class=" mx-5">
157 <nav class="navbar navbar-expand-lg navbar-light p-0">
158 <a class="navbar-brand" href="../view/homePage.php" id="logo">
159 
160 </a>
161 <div class="collapse navbar-collapse" id="navbarSupportedContent">
162 <ul class="navbar-nav mr-auto" style="margin: auto;">
163 <li class="nav-item active">
164 <?php echo $navMenu1?>
165 </li>
166 <li class="nav-item active">
167 <?php echo $navMenu2?>
168 </li>
169 </ul>
170 <form class="form-inline my-2 my-lg-0" method="get" action="../view/search.php">
171 <input class="form-control mr-sm-0" type="search" placeholder="Rechercher" ↵
172 aria-label="Search" id="searchBar" name="nameArtistOrTitleTab">
173 <button type="submit" class="btn btn-outline-secondary ">
174 <i class="fas fa-search"></i>
175 </button>
176 </form>
177 </div>
178 </div>
179 </div>
180 <?php echo $body; ?>
181 <script src="../public/js/function.js"></script>
182 <script type="text/javascript">
183 $( document ).ready(function() {
184 var idTab = <?php echo $_GET['idTab'];?>;
185 var idUserComment = <?php if(isset($userId)){echo $userId;}else{echo -1;} ?>;
186 var idTabComment = <?php if(isset($idTab)){echo $idTab;}else{echo -1;}?>;
187 var pseudoUser = "<?php if(isset($pseudoUser)){echo $pseudoUser;}else{echo "";} ?>";
188 get_data("../controller/getCommentsByTab.php",getCommentsByTab,{ 'idTab': idTab,true);
189 function getCommentsByTab(data) {
190 data.forEach(function (comment) {
191 <b></b></label><label> '+comment.contentComment+'</label></td></tr>')
192 });
193 }
194
195 get_data("../controller/getRateByTabId.php",getRateByTabId,{ 'idTab': idTab,true);
196 function getRateByTabId(data)
197 {
198 data.forEach(function (rate) {
199 if (rate.users_idUsers == idUserComment) {
200 <?php echo $rate.rate;?>
201 }
202 });
203 }
204
205 get_data("../controller/getTabById.php",getTabById,{ 'idTab': idTab,true);
206 function getTabById(data) {
207 data.forEach(function (tab) {
208 if (tab.rateTab > 0)
209 {
210 <?php echo $tab.rateTab;?>
211 }
212 else
213 {
214 <?php echo "La tablature n'est pas notée";?>
215 }
216 });
217 }
218
219 }

```

```

220
221
222
223     $('#postComment').click(function () {
224
225
226         var contentComment = $('#myComment').val();
227
228
229         if(idUserComment != -1 && idTabComment != -1)
230         {
231             get_data("../controller/addComment.php", addComment, {
232                 'contentComment': contentComment,
233                 'idTabComment': idTabComment,
234                 'idUserComment': idUserComment
235             }, true);
236
237             function addComment(data) {
238
239                 if (data == true) {
240                     alert("votre commentaire a bien été enregistré !");
241                     location.reload();
242                 }
243                 else {
244                     alert("un problème est survenu");
245                 }
246             }
247         }
248         else
249         {
250             alert("vous devez être connecté pour laisser une appréciation");
251         }
252     });
253
254     $('#btnAddUser').click(function () {
255         var name = $('#name').val();
256         var forename = $('#forename').val();
257         var password = $('#password').val();
258         var passwordConfirm = $('#passwordConfirm').val();
259         var email = $('#email').val();
260         var pseudo = $('#pseudo').val();
261         get_data("../controller/getUserByPseudo.php",getUserByPseudo,{ 'pseudo' : pseudo},true);
262         function getUserByPseudo(dataPseudo)
263         {
264             if(dataPseudo == false)
265             {
266                 get_data("../controller/getUserByEmail.php",getUserByEmail,{ 'email' : email},true);
267                 function getUserByEmail(dataEmail)
268                 {
269                     if(dataEmail == false)
270                     {
271                         get_data("../controller/addUser.php",addUser,{ 'name' :name, 'forename' : ↵
forename, 'password' : password, 'email' : email, 'pseudo' : pseudo, 'passwordConfirm' : ↵
passwordConfirm},true);
272                         function addUser(){
273                             identifyUser(pseudo,password);
274                             $('#addUser').modal('hide');
275                         }
276                     }
277                 }
278                 else
279                 {
280                     alert('Ce mail est déjà pris, veuillez en utiliser un autre');
281                 }
282             }
283         }
284     }
285     else
286     {
287         alert('Ce pseudo est déjà pris, veuillez en utiliser un autre');
288     }
289 }
290 }
291 });
292
293
294     $('#btnIdentifyUser').click(function () {
295         var mailOrPseudo = $('#pseudoOrMail').val();
296         var pwdConnexion = $('#pwdConnexion').val();
297         identifyUser(mailOrPseudo,pwdConnexion);
298         $('#connectUser').modal('hide');
299
300     });
301
302     $('#rateStar').mouseenter(function () {
303         for(i = $(this).attr('id');i>=1;i--)
304         {
305             $('#'+i).removeClass("far fa-star");
306             $('#'+i).addClass("fas fa-star");
307         }
308
309     });
310
311     $('#rateStar').mouseleave(function () {

```

```

312         $(this).removeClass("fas fa-star");
313         $(this).addClass("far fa-star");
314     });
315
316     $('#rateStar').click(function () {
317
318         if(idUserComment != -1) {
319             var rate = $(this).attr('id');
320
321             get_data("../controller/addRate.php", addRate, {
322                 'idUser': idUserComment,
323                 'idTab': idTab,
324                 'rate': rate
325             }, true);
326
327             function addRate(data) {
328                 if (data == true && pseudoUser != "") {
329
330                     alert("votre note a bien été enregistrée !");
331                     location.reload();
332                 }
333                 else {
334                     alert("un problème est survenu");
335                 }
336             }
337         }
338         else
339         {
340             alert("vous devez être connecté pour laisser une appréciation");
341         }
342     });
343
344 });
345 </script>

```

1.6.4 tablatureManagerPage.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 16.05.2018
5  * projet: SimpleTab
6  * description : page de gestion de l'utilisateur
7  */
8
9 session_start();
10
11 require_once "../view/modals.php";
12
13 $navIdOrButton="";
14 $navMenu1 = "";
15 $navMenu2 = "";
16 $redirect = 1;
17
18 if(isset($_SESSION['user']))
19 {
20     $navIdOrButton = "<div class=\"dropdown\">
21         <button type=\"button\" onclick=\"\" class=\"btn btn-light dropdown-toggle\" ↵
22         data-toggle=\"dropdown\" aria-haspopup=\"true\" aria-expanded=\"false\"> ↵
23         \"$_SESSION['user'][0]['pseudoUser']\"</button>
24         <div class=\"dropdown-menu\" aria-labelledby=\"dropdownMenuButton\">
25             <a class=\"dropdown-item\" ↵
26             href=\"../controller/destroySession.php\">Déconnexion</a>
27         </div>
28     </div>";
29     if($_SESSION['user'][0]['role_idrole'] == 0)
30     {
31         $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
32         class=\"sr-only\">(current)</span></a></h5>";
33         $navMenu2 = "<h5><a class=\"nav-link\" href=\"../view/tablatureManagerPage.php\">Gestion des ↵
34         tablatures </a></h5>";
35     }
36     elseif($_SESSION['user'][0]['role_idrole'] == 1)
37     {
38         $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
39         class=\"sr-only\">(current)</span></a></h5>";
40         $navMenu2 = "<h5><a class=\"nav-link\" href=\"../view/tablatureAndUserManagerPage.php\">Gestion des ↵
41         tablatures et utilisateurs </a></h5>";
42     }
43 }
44 else
45 {
46     $redirect = 0;
47 }
48 ?>
49 <html>
50 <head>
51     <title>Gestion des tablatures</title>
52     <script src="https://code.jquery.com/jquery-3.3.1.js" ↵
53     integrity="sha256-2Kok7Mb0yxpqUVvAk/HJ2jigOSYS2auK4Pfbzm7uH60=" crossorigin="anonymous"></script>

```

```

46 <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" ↵
    integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPiPM49" ↵
    crossorigin="anonymous"></script>
47 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" ↵
    integrity="sha384-smHYKdLADwkX01EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQpOMk8ck+5T" ↵
    crossorigin="anonymous"></script>
48 <!-- Copyright (c) 2016 Indri Muska. Licensed under the MIT license.-->
49 <script ↵
    src="//raw.githubusercontent.com/indrimuska/jquery-editable-select/master/dist/jquery-editable-select.min.js"></script>
50 <link ↵
    href="//raw.githubusercontent.com/indrimuska/jquery-editable-select/master/dist/jquery-editable-select.min.css" ↵
    rel="stylesheet">
51
52 <link rel="stylesheet" type="text/css" href="../../public/css/style.css">
53 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css" ↵
    integrity="sha384-9gVQ44YFwwWSjIDZnLEWnxCjeSFWFphJiwGPXr1jddIh0egiu1Fw05qRgVFX0dJZ4" ↵
    crossorigin="anonymous">
54
55 <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.0.13/css/all.css" ↵
    integrity="sha384-DNOH268U8hZfKX0rtjWvJxusGo9WQnrNx2sqG0tfsghAvtVRW3tvkXWZ58N9jp" ↵
    crossorigin="anonymous">
56 <!-- JQuery CDN -->
57 </head>
58 <body>
59 <div class="border">
60 <div class="col text-right mr-3 mb-0"> <?php echo $navIdOrButton;?>
61
62 <div id="body" class="mx-5">
63 <nav class="navbar navbar-expand-lg navbar-light p-0">
64 <a class="navbar-brand" href="../../view/homePage.php" id="logo">
65 
66 </a>
67
68 <div class="collapse navbar-collapse" id="navbarSupportedContent">
69 <ul class="navbar-nav mr-auto" style="margin: auto;">
70 <li class="nav-item active">
71 <?php echo $navMenu1?>
72 </li>
73 <li class="nav-item active">
74 <?php echo $navMenu2?>
75 </li>
76 </ul>
77 <form class="form-inline my-2 my-lg-0" method="get" action="../../view/search.php">
78 <input class="form-control mr-sm-0" type="search" placeholder="Rechercher" ↵
    aria-label="Search" id="searchBar" name="nameArtistOrTitleTab">
79 <button type="submit" class="btn btn-outline-secondary">
80 <i class="fas fa-search"></i>
81 </button>
82 </form>
83 </div>
84 </nav>
85 </div>
86 </div>
87 </div>
88 <div id="message" style="display:none;">
89 </div>
90
91 <div class="m-5">
92
93 <table class="table table-dark">
94 <thead style="color:red">
95 <tr class="text-center">
96 <th colspan="6"><button class="btn btn-dark" data-toggle="modal" data-target="#addTab" ↵
    style="color:red;">Ajouter une tablature</button></a></i></th>
97 </tr>
98 <tr>
99 <th scope="col">Artiste</th>
100 <th scope="col">Titre</th>
101 <th scope="col">Difficulté</th>
102 <th scope="col">Note</th>
103 <th scope="col" colspan="2" style="text-align: center;">Gestion</th>
104 </tr>
105 </thead>
106 <tbody id='tabs'>
107 </tbody>
108 </table>
109 </td>
110
111
112 </div>
113 </body>
114 <script src="../../public/js/function.js"></script>
115 <script type="text/javascript">
116 $( document ).ready(function() {
117
118 var redirect = <?php echo $redirect;?>;
119
120 if(redirect == 1 ) {
121
122
123 var idUser = <?php if(isset($_SESSION['user'] [0] ['idUser'])) { echo ↵
    $_SESSION['user'] [0] ['idUser']; } else {echo -1; }?>;
124 reloadTabByUSers(idUser);
125

```

```

126 get_data("../controller/getArtistNames.php",getArtistNames,{},true);
127 function getArtistNames(data)
128 {
129     data.forEach(function (artist) {
130         $('#addArtist').append('<option>'+artist.nameArtist+'</option>');
131         $('#modifyAuthor').append('<option>'+artist.nameArtist+'</option>');
132     })
133 }
134
135 $('#addArtist').editableSelect({ effects: 'default' });
136 $('#modifyAuthor').editableSelect({ effects: 'default' });
137
138
139 $('#btnAddTab').click(function () {
140     var title = $('#addTitle').val();
141     var artist = $('#addArtist').val();
142     var lvl = $('#addLvl option:selected').val();
143     var capo = $('#addCapo').val();
144     var key = $('#addKey').val();
145     var tuning = $('#addTuning').val();
146     var tabBody = $('#addTablatureBody').val();
147     get_data("../controller/addTab.php", addTab, {
148         'addTitle': title,
149         'addArtist': artist,
150         'addLvl': lvl,
151         'addCapo': capo,
152         'addKey': key,
153         'addTuning': tuning,
154         'addTabBody': tabBody,
155     }, false);
156
157     function addTab(data) {
158         var message = "";
159         if (data == true) {
160             alert("La tablature a bien été ajoutée, elle sera visible quand l'administrateur ←
161 l'aura acceptée")
162         }
163         else {
164             alert("Un problème est survenu")
165         }
166         reloadTabByUSers(idUser);
167
168     }
169
170     $('#addTab').modal('hide')
171     reloadTabByUSers(idUser);
172 });
173
174
175
176 $('#btnModifyTab').click(function () {
177     var title = $('#modifyTitle').val();
178     var artist = $('#modifyAuthor').val();
179     var lvl = $('#modifyLvl option:selected').val();
180     var capo = $('#modifyCapo').val();
181     var key = $('#modifyKey').val();
182     var tuning = $('#modifyTuning').val();
183     var tabBody = $('#modifyTablatureBody').val();
184     get_data("../controller/modifyTabById.php", modifyTabById, {
185         'modifyTitle': title,
186         'modifyAuthor': artist,
187         'modifyLvl': lvl,
188         'modifyCapo': capo,
189         'modifyKey': key,
190         'modifyTuning': tuning,
191         'modifyTabBody': tabBody,
192     }, false);
193
194     function modifyTabById(data) {
195         if (data == true) {
196             alert('La tablature a été modifiée avec succès');
197         }
198         else {
199             alert("Un problème est survenu")
200         }
201         reloadTabByUSers(idUser);
202
203     }
204
205     $('#modifyTab').modal('hide')
206     reloadTabByUSers(idUser);
207 });
208
209 }
210 }
211 else
212 {
213     window.location.href = "../view/homePage.php";
214 }
215 });
216 </script>

```

1.6.5 modals.php

```
1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 11.05.2018
5  * projet: SimpleTab
6  * description : modals du site
7  */
8
9 ?>
10 <!-- Modal Pop-up addUser() -->
11 <div class="modal fade" id="addUser" role="dialog">
12   <div class="modal-dialog modal-dialog-centered">
13
14     <!-- Modal content-->
15     <div class="modal-content ">
16       <div class="modal-header pb-0">
17         <table>
18           <tr>
19             <td>
20               <h4 class="modal-title " style="text-align: left;">Inscription</h4>
21             </td>
22           </tr>
23           <tr>
24             <td> <small class="text-muted"> Les champs marqués d'un <label style="color: ↵
25 red;">*</label> sont obligatoires</small></td>
26           </tr>
27         </table>
28
29       </div>
30
31       <div class="modal-body">
32         <table class="col">
33           <tr>
34             <td> <label class="text-left">Nom <label style="color: red;">*</label> : ↵
35             <td class="float-right"> <input type="text" class=" emptyForbiddenField" id="name" ↵
36               style="resize: none;" autofocus ></td>
37             </tr>
38             <td><label class="text-left">Prénom <label style="color: red;">*</label> : ↵
39             <td class="float-right"><input type="text" class=" emptyForbiddenField" ↵
40               id="forename" style="resize: none;" ></td>
41             </tr>
42             <td><label class="text-left">Pseudo<label style="color: red;">*</label> : ↵
43             <td class="float-right"><input type="text" class=" emptyForbiddenField" ↵
44               id="pseudo" style="resize: none;" ></td>
45             </tr>
46             <td><label class="text-left">Email <label style="color: red;">*</label> : ↵
47             <td class="float-right"><input type="email" class=" emptyForbiddenField" ↵
48               id="email" style="resize: none;" ></td>
49             </tr>
50             <td><label class="text-left">Mot de passe <label style="color: red;">*</label> : ↵
51             <td class="float-right"> <input type="password" class=" emptyForbiddenField" ↵
52               id="password" style="resize: none;" ></td>
53             </tr>
54             <td><label class="text-left">Retapez le mot de passe <label style="color: ↵
55 red;">*</label> : </label></td>
56             <td class="float-right"><input type="password" class=" emptyForbiddenField" ↵
57               id="passwordConfirm" style="resize: none;" ></td>
58             </tr>
59             <td colspan="2"><button style="color:red;" id="btnAddUser" type="button" class="btn ↵
60               btn-dark col">Rejoindre SimpleTab</button></td>
61             </tr>
62           </table>
63         </div>
64         <!-- Modal footer-->
65         <div class="modal-footer">
66           <table class="col text-center">
67             <tr>
68               <td><label> Déjà inscrit ? </label></td>
69             </tr>
70             <tr>
71               <td>
72                 <button type="button" class="btn btn-light" data-toggle="modal" ↵
73                 data-target="#connectUser" onclick="hideModal()">Identifiez-vous !</button>
74               </td>
75             </tr>
76           </table>
77         </div>
78       </div>
79     </div>
80   </div>
```

```

77 </div>
78 <!-- END Modal -->
79
80
81 <!-- Modal Pop-up connectUser() -->
82 <div class="modal fade" id="connectUser" role="dialog">
83   <div class="modal-dialog modal-dialog-centered">
84     <!-- Modal content-->
85     <div class="modal-content">
86       <div class="modal-header pb-0">
87         <h4 class="modal-title " style="text-align: left;">Connexion</h4>
88       </div>
89       <div class="modal-body">
90         <table class="col">
91           <tr>
92             <td> <label class="text-left">Pseudo ou email: </label></td>
93             <td class="float-right"> <input type="text" class="mb-2 emptyForbiddenField" ↵
94               id="pseudoOrMail" style="resize: none;" autofocus></td>
95           </tr>
96           <tr>
97             <td><label class="text-left">Mot de passe: </label></td>
98             <td class="float-right"><input type="password" class=" emptyForbiddenField" ↵
99               id="pwdConnexion" style="resize: none;" ></td>
100           </tr>
101           <tr >
102             <td colspan="2"><button style="color:red;" id="btnIdentifyUser" type="button" ↵
103               class="btn btn-dark col mt-2">Connexion</button></td>
104           </tr>
105         </table>
106       </div>
107     </div>
108   </div>
109 </div>
110 <!-- END Modal -->
111
112 <!-- Modal Add Tab Form -->
113 <div class="modal fade " id="addTab" role="dialog">
114   <div class="modal-dialog modal-dialog-centered modal-lg" style="display: table;">
115     <!-- Modal content-->
116     <div class="modal-content">
117       <div class="modal-header pb-0">
118         <h4 class="modal-title " style="text-align: left;">Ajouter une tablature</h4>
119       </div>
120       <div class="modal-body">
121         <table class="col">
122           <tr>
123             <td><label class="text-left">Titre: </label></td>
124             <td><label class="text-left">Auteur: </label></td>
125             <td><label class="text-left">Difficulté: </label></td>
126             <td><label class="text-left">Capo: </label></td>
127             <td><label class="text-left">Tonalité: </label></td>
128             <td><label class="text-left">Accordage: </label></td>
129           </tr>
130           <tr>
131             <td class="pr-2"> <input type="text" class=" emptyForbiddenField" id="addTitle" ↵
132               style="resize: none;" autofocus></td>
133             <td class="pr-2"> <select type="text" class=" emptyForbiddenField" id="addArtist" ↵
134               style="resize: none;" >
135               </select></td>
136             <td class="pr-2" style="display: inline"><select class="form-control" id="addLvl">
137               <option value="0">Facile</option>
138               <option value="1">Moyen</option>
139               <option value="2">Difficile</option>
140             </select></td>
141             <td class="pr-2"> <input type="text" class=" emptyForbiddenField" id="addCapo" ↵
142               style="resize: none;" ></td>
143             <td class="pr-2"> <input type="text" class=" emptyForbiddenField" id="addKey" ↵
144               style="resize: none;" ></td>
145             <td class="pr-2"> <input type="text" class=" emptyForbiddenField" id="addTuning" ↵
146               style="resize: none;" value="E A D G B E"></td>
147           </tr>
148           <tr>
149             <td colspan="6"><div>
150               <div>Tablature</div>
151               <div>
152                 <div>
153                   <div>
154                     <div>
155                       <div>
156                         <div>
157                           <div>
158                             <div>
159                               <div>
160                               </div>
159                             </div>
160                             </div>
159                           </div>
158                         </div>
157                       </div>
156                     </div>
155                   </div>
154                 </div>
153               </div>
152             </td>
151             <td colspan="6"><button style="color:red;" id="btnAddTab" type="button" class="btn ↵
150               btn-dark col mt-2">Ajouter</button></td>
151           </tr>
152         </table>
153       </div>
154     </div>
155   </div>
156 </div>
157 </div>
158 </div>
159 </div>
160 </div>

```



```

161 <!-- END Modal -->
162
163 <!-- Modal Modify Tab Form -->
164 <div class="modal fade bd-example-modal-lg" id="modifyTab" role="dialog">
165   <div class="modal-dialog modal-dialog-centered modal-lg" style="display: table;">
166
167     <!-- Modal content-->
168     <div class="modal-content">
169       <div class="modal-header pb-0">
170
171         <h4 class="modal-title " style="text-align: left;">Modifier une tablature</h4>
172       </div>
173       <div class="modal-body">
174         <table class="col">
175           <tr>
176             <td> <label class="text-left">Titre: </label></td>
177             <td><label class="text-left">Auteur: </label></td>
178             <td><label class="text-left">Difficulté: </label></td>
179             <td><label class="text-left">Capo: </label></td>
180             <td><label class="text-left">Tonalité: </label></td>
181             <td><label class="text-left">Accordage: </label></td>
182           </tr>
183           <tr>
184             <td class="pr-2"> <input type="text" class=" emptyForbiddenField" id="modifyTitle" ↵
185               style="resize: none;" autofocus></td>
186             <td class="pr-2"> <select type="text" class=" emptyForbiddenField" ↵
187               id="modifyAuthor" style="resize: none;" >
188               </select></td>
189             <td class="pr-2"><select class="form-control" id="modifyLvl">
190               <option value="0">Facile</option>
191               <option value="1">Moyen</option>
192               <option value="2">Difficile</option>
193             </select></td>
194             <td class="pr-2"> <input type="text" class=" emptyForbiddenField" id="modifyCapo" ↵
195               style="resize: none;" ></td>
196             <td class="pr-2"> <input type="text" class=" emptyForbiddenField" id="modifyKey" ↵
197               style="resize: none;" ></td>
198             <td class="pr-2"> <input type="text" class=" emptyForbiddenField" ↵
199               id="modifyTuning" style="resize: none;" ></td>
200           </tr>
201           <tr>
202             <td colspan="6">Tablature</td>
203           </tr>
204           <tr>
205             <td colspan="6"><div id="modifyTablatureBody"></div></td>
206           </tr>
207           <tr>
208             <td colspan="6"><button style="color:red;" id="btnModifyTab" type="button" ↵
209               class="btn btn-dark col mt-2">Modifier</button></td>
210           </tr>
211         </table>
212       </div>
213     </div>
214   </div>
215 </div>
216 <!-- END Modal -->

```

1.6.6 tablatureAndUserManagerPage.php

```

1 <?php
2 /**
3  * Auteur: romain.ssr@eduge.ch
4  * Date : 22.05.2018
5  * projet: SimpleTab
6  * description : page de gestion de l'administrateur
7  */
8
9
10 session_start();
11
12 require_once "modals.php";
13
14 $navIdOrButton = "";
15 $navMenu1 = "";
16 $navMenu2 = "";
17 $redirect = 1;
18
19
20 if(isset($_SESSION['user']))
21 {
22   $navIdOrButton = "<div class=\"dropdown\">
23     <button type=\"button\" onclick=\"\" class=\"btn btn-light dropdown-toggle\" ↵
24     data-toggle=\"dropdown\" aria-haspopup=\"true\" aria-expanded=\"false\"> ↵
25     \"._SESSION['user'][0]['pseudoUser']\"</button>
26     <div class=\"dropdown-menu\" aria-labelledby=\"dropdownMenuButton\">
27       <a class=\"dropdown-item\" ↵
28       href=\"../controller/destroySession.php\">Déconnexion</a>

```



```

26         </div>
27     </div>";
28     if($_SESSION['user'][0]['role_idrole'] == 0)
29     {
30         $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
31         class=\"sr-only\">(current)</span></a></h5>";
32         $navMenu2 = "<h5><a class=\"nav-link\" href=\"../view/tablatureManagerPage.php\">Gestion des ↵
33         tablatures </a></h5>";
34     }
35     elseif($_SESSION['user'][0]['role_idrole'] == 1)
36     {
37         $navMenu1 = "<h5><a class=\"nav-link\" href=\"../view/homePage.php\">Accueil <span ↵
38         class=\"sr-only\">(current)</span></a></h5>";
39         $navMenu2 = "<h5><a class=\"nav-link\" href=\"../view/tablatureAndUserManagerPage.php\">Gestion des ↵
40         tablatures et utilisateurs </a></h5>";
41     }
42 }
43 else
44 {
45     $redirect = 0;
46 }
47 ?>
48 <!DOCTYPE HTML>
49 <html>
50 <head>
51     <title>Accueil</title>
52     <script src="https://code.jquery.com/jquery-3.3.1.js" ↵
53     integrity="sha256-2Kok7Mb0yxpGUvAk/HJ2jig0SYS2auK4Pfzbm7uH60=" crossorigin="anonymous"></script>
54     <script src="https://cdn.jsdelivr.net/npm/popper.js/1.14.3/umd/popper.min.js" ↵
55     integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/18WvCWPiPM49" ↵
56     crossorigin="anonymous"></script>
57     <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" ↵
58     integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQpOMk8ck+5T" ↵
59     crossorigin="anonymous"></script>
60     <link rel="stylesheet" type="text/css" href="../public/css/style.css">
61     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css" ↵
62     integrity="sha384-9gVQ4dYFwwWSjIDZnLEWnxCjeSWFphJiWGPXr1jddIhOegui1Fw05qRGrVFX0dJZ4" ↵
63     crossorigin="anonymous">
64     <!-- JQuery CDN -->
65 </head>
66 <body>
67 <div class="border ">
68     <div class="col text-right mr-3 mb-0"> <?php echo $navIdOrButton;?>
69
70     <div id="body" class="mx-5">
71         <nav class="navbar navbar-expand-lg navbar-light p-0">
72             <a class="navbar-brand" href="../view/homePage.php" id="logo">
73                 
74             </a>
75
76             <div class="collapse navbar-collapse" id="navbarSupportedContent">
77                 <ul class="navbar-nav mr-auto" style="margin: auto;">
78                     <li class="nav-item active">
79                         <?php echo $navMenu1?>
80                     </li>
81                     <li class="nav-item active">
82                         <?php echo $navMenu2?>
83                     </li>
84                 </ul>
85
86                 <form class="form-inline my-2 my-lg-0" method="get" action="../view/search.php">
87                     <input class="form-control mr-sm-0" type="search" placeholder="Rechercher" ↵
88                     aria-label="Search" id="searchBar" name="nameArtistOrTitleTab">
89                     <button type="submit" class="btn btn-outline-secondary ">
90                         <i class="fas fa-search"></i>
91                     </button>
92                 </form>
93             </div>
94         </nav>
95     </div>
96 </div>
97
98 <div class="m-5">
99     <table class="table table-dark">
100         <thead style="color:red">
101             <tr>
102                 <th colspan="4" class="text-center">Gestion des tablatures</th>
103             </tr>
104             <tr>
105                 <th scope="col">Artiste</th>
106                 <th scope="col">Titre</th>
107                 <th scope="col">Difficulté</th>
108                 <th scope="col">Note</th>
109             </tr>
110         </thead>
111         <tbody id='tabs'>

```

```

106     </tbody>
107 </table>
108 <table class="table table-dark">
109   <thead style="color:red">
110     <tr>
111       <th colspan="4" class="text-center">Gestion des utilisateurs</th>
112     </tr>
113     <tr>
114       <th scope="col">Pseudo</th>
115       <th scope="col">Email</th>
116       <th scope="col">Nombre de tablatures ajoutées</th>
117       <th scope="col">Gestion</th>
118     </tr>
119   </thead>
120   <tbody id='users'>
121   </tbody>
122 </table>
123 </div>
124 </body>
125 <script src="../../public/js/function.js"></script>
126 <script type="text/javascript">
127   $( document ).ready(function() {
128
129     var redirect = <?php echo $redirect ?>
130
131     if(redirect == 1) {
132
133       getAllNonApprouvedTabs();
134
135       getUsersAndNbTabPosted();
136
137
138     }
139     else
140     {
141       window.location.href = "../../view/homePage.php";
142     }
143
144   }
145   });
146 </script>
147

```